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]

Presented by: M Harshvardhan

[1]Alibaba Group, China

[2]The University of Tennessee, Knoxville

What is Guaranteed Delivery Advertisement?

Background of Online Advertisement

➔**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

➔**Key features of GD Advertising:**

- ◆ Fixed price for impressions
- ◆ Defined target audience
- ◆ Specified target areas
- ◆ Chosen target channels

➔**Advantages of GD over Real-time Bidding**

- ◆ Compared to the dynamic and competitive nature of RTB, GD offers a cost-effective solution for brand advertisers
- ◆ GD ensures wide reach to potential consumers with guaranteed impressions, providing better control over advertising campaigns and budget

Advertising Allocation Strategies

- Guaranteed Delivery (GD)
- Real-time Bidding (RTB)

Background and Contributions

How does Guaranteed Delivery Allocation work, how does our end-to-end model work, and what are our contributions?

We visualize End-to-End Prediction and Allocation System

System Architecture for Allocating Ads at Alibaba

➔**Our focus is Selling Systems.** The goal is to establish contracts with advertisers in advance by predicting and allocating inventory accurately

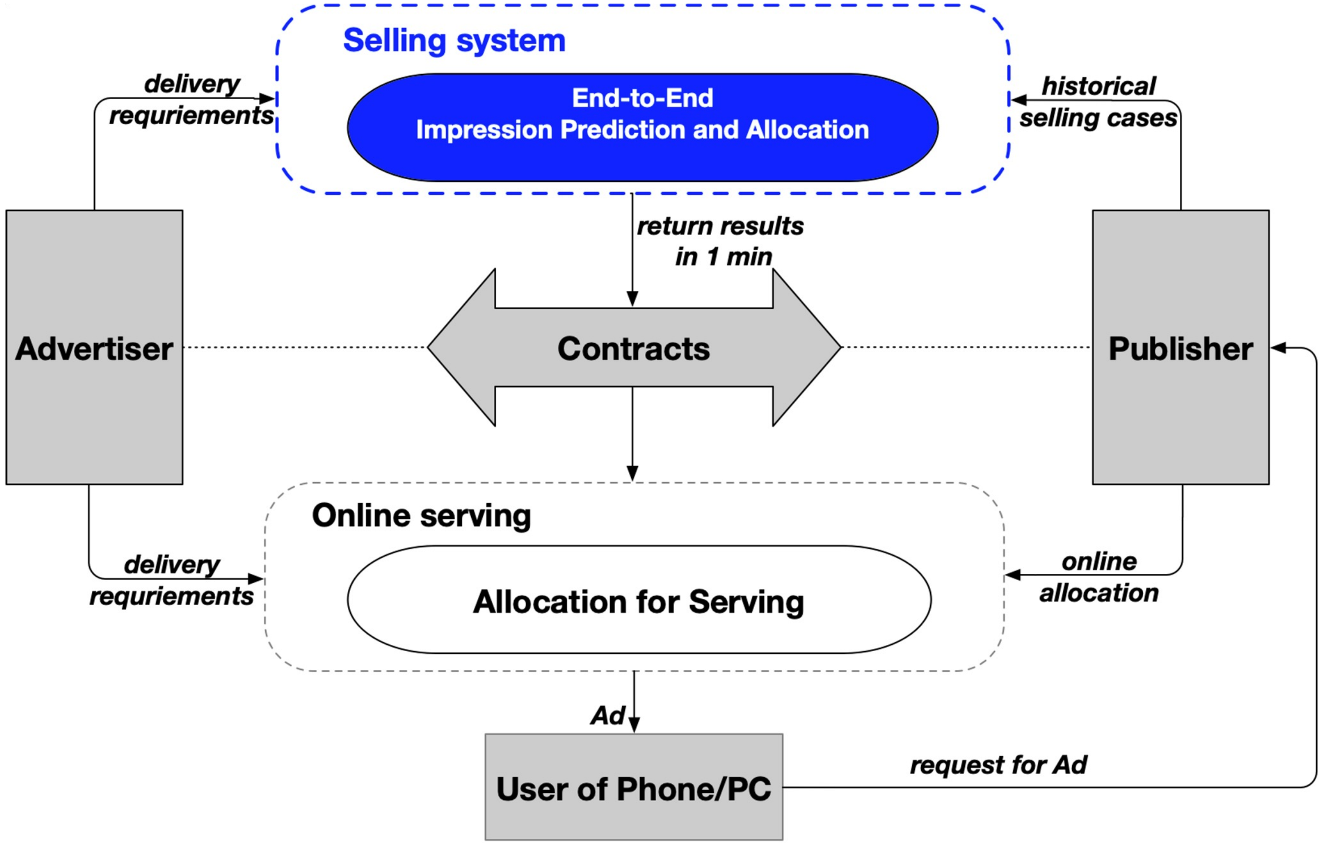
➔We sign contracts with advertisers in advance while having limited impressions inventory

➔The objectives are to:

- Maximize inventory sales
- Prevent overselling of inventory

➔**Online Serving System** ensures

- Fulfillment of reserved contracts
- Click-Through Rate (CTR) Optimization



The system architecture for Guaranteed Delivery (GD) advertising.
Our model supports the selling system, when advertisers and the publisher sign new GD contracts.

Traditional methods focus on real-time dispatching of traffic to ads

Overview of Traditional Two-stage Systems

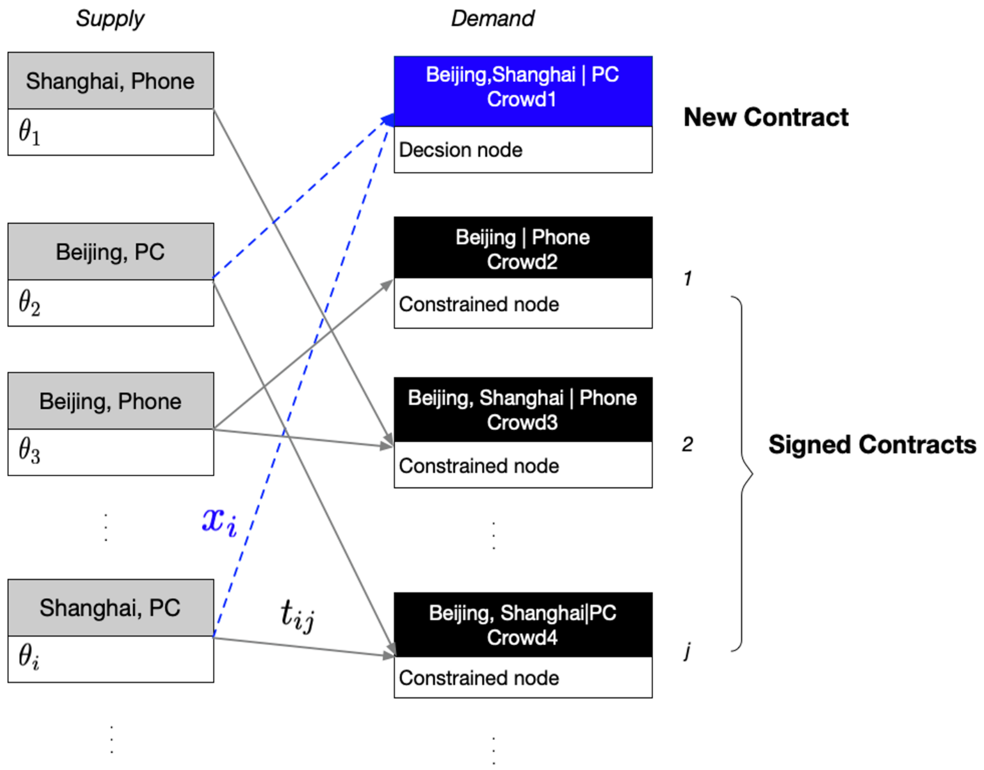


Figure: Bipartite graph of contract allocation problem. Supply nodes are impression inventories while demand nodes are impression contracts.

Real-Time Bidding (RTB):
An auction-based system where ad impressions are bought and sold on a per-case basis. Advertisers bid for an impression, and if they win, their ad is instantly displayed.

- **Pros:** Dynamic, allows for real-time customization and targeting
- **Cons:** Uncertain costs, less control over ad placement

Traditional Guaranteed Delivery (GD) Systems:
Advertisers sign contracts with publishers in advance to secure a fixed inventory of ad impressions.

- **Pros:** Secured impressions, cost control
- **Cons:** Less flexibility, may lead to underutilization or overselling of inventory

Given that high-quality impressions often sell out swiftly in large-scale GD marketplaces, even a marginal improvement in inventory usage can significantly increase publisher revenue and serve more advertisers.

Closing the Gap: Two-Stage Systems vs. Our End-to-End Model

What are the limitations and how we bridge it?

Limitations of Two-Stage Systems

Lack of Real-Time Support

- Traditional studies on GD advertising don't provide real-time predictive allocations required for advance contract signing

Overselling Risk

- Theoretical models for online serving risk overselling if the signed target impressions exceed the inventory limit

Forecasting Uncertainty

- Two-stage methods assume lower forecasting error translates to better allocation quality, which isn't always the case

Inefficient Handling of Constraints

- Existing end-to-end learning-based optimization frameworks struggle with large numbers of constraints in GD selling.

Neural Lagrangian Selling (NLS)

End-to-End Approach

- We propose an end-to-end approach that integrates inventory prediction and contract allocation, overcoming the limitations of two-stage methods

Efficient Solver with Less Memory

- Our GD selling system includes an efficient differentiable Lagrangian solver with less memory cost

Dynamic Feature Extraction

- We use Graph Convolutional Networks (GCNs) to capture dynamic features from advertising contracts, enabling NLS to fit dynamic optimization objectives and constraints

Improved Performance

- Our approach provides improved allocation error, demonstrating its effectiveness in real-world GD selling systems

Methodology

Deep-dive into how our End-to-End Allocation optimization model works

End-to-End Model Minimizes Allocation Regret Instead of Prediction Loss

Stage 1: Impression Inventory

- Predictive model (like NN) to forecast impressions
- Let θ be the impression inventory of supply nodes; z has historical and contextual information
- Predict impressions by minimizing loss

$$Loss = ||\theta - g(z; w)||^2$$

Stage 2: Inventory Allocation

- θ is supply inventory, t represents the elements of allocation matrix, p are penalty constraints (importance of different constraints), u are slack variables
- $\max_{\{x,t,u\}} (I \odot \theta)'x - p'u,$
- Subject to constraints:

$$Gx + Bt - u \leq h,$$
$$u \geq 0,$$
$$x, t \in [0,1]$$

Our End-to-End Approach

- NLS minimizes allocation regret, not just prediction loss
- Learns from historical decision cases, considering both inventory prediction and allocation

Allocation Regret

- Formally defined as $\text{Regret} = \|(I \odot \hat{\theta})^\top \hat{x} - (I \odot \theta)^\top x\|_2^2$
- $\hat{\theta} = g(z; w)$ represents predicted inventories, $\hat{x}$ corresponds to allocation decisions

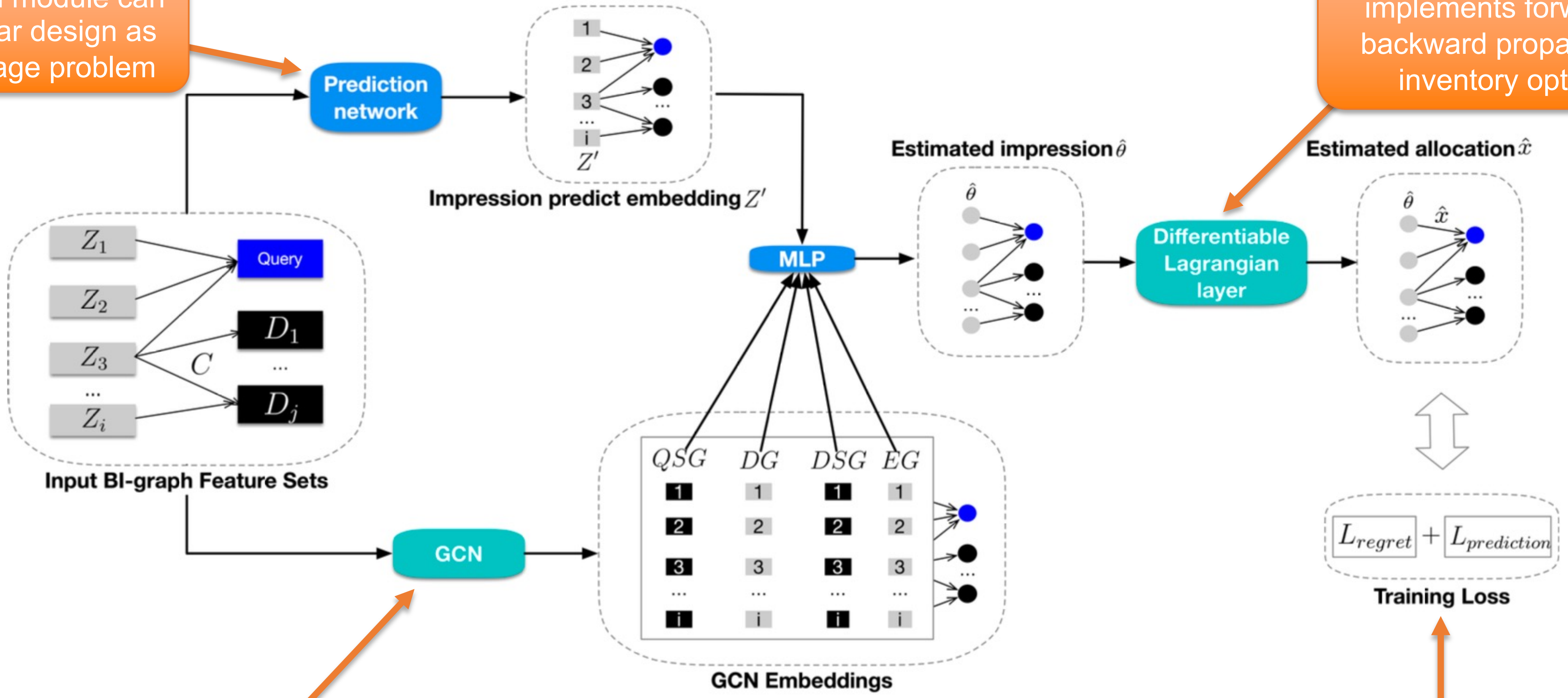
Gradient of Regret

- Gradient of the regret can be computed by differentiating the allocation optimization problem
- Challenge lies in backpropagating optimum $\frac{\partial \hat{x}}{\partial \hat{\theta}}$

Architecture of Neural Lagrangian Selling (NLS) Model

Prediction module can use similar design as in two-stage problem

Lagrangian module implements forward and backward propagation of inventory optimizer



GCN module extracts important information from the allocation bipartite graph

Allocation regret instead of prediction loss

Lagrangian Dual Optimization

- We formulate dual problem to minimize

$$\min_{\alpha,\beta\geq 0} \max_{x,t,u} L(x,t,u,\alpha,\beta),$$

- where the Lagrangian function is

$$L(x,t,u,\alpha,\beta) = \ell(x,t,u) + \alpha^\top(h + u - Gx - Bt) + \beta^\top u.$$

- We derive corresponding KKT (Karush-Kuhn-Tucker) conditions:

$$\begin{aligned} x &= \min(\max(\frac{I \odot \theta - G^\top \alpha}{\lambda}, 0), 1), \\ t &= \min(\max(\frac{-B^\top \alpha}{\lambda}, 0), 1), \\ \alpha \odot (h + u - Gx - Bt) &= 0, \\ \beta \odot u &= 0, \\ \alpha + \beta &= p. \end{aligned}$$

- When $u > 0$, then $\beta = 0$ and $\alpha = p$.
- When $u = 0$, the gradient is obtained as:

$$\begin{aligned} \frac{\partial L}{\partial \alpha} =& (I \odot \theta - \lambda x - G^\top \alpha)\frac{\partial x}{\partial \alpha} - (B^\top \alpha + \lambda t)\frac{\partial t}{\partial \alpha} \\ &+ (h - Gx - Bt), \end{aligned}$$

- We choose hyperparameter $\lambda = \min(\theta)$

- Small λ results in instability while large λ introduces errors

Efficient Lagrangian Dual Layer

Integrating the Lagrangian Dual Optimization Layer in Deep Neural Networks

Solving the GD Selling Allocation Problem

- We integrate the Lagrangian dual optimization layer into predictive deep neural networks, as outlined in *Algorithm 1*.
- Adam gradient descent method is used in Stage 3, with $b1, b2, \mu$ and $decay\_rate$ as hyperparameters

Advantages of Our Approach

- **Flexibility:** Our Lagrangian dual layer can be seamlessly embedded into any neural network structure.
- **Autonomy:** Back-propagation can be performed automatically by deep learning frameworks like TensorFlow or Torch.
- **Efficiency:** The computation of the optimization layer avoids complex operations like matrix inversion, enabling easy implementation with batched input and parallel computing for enhanced training/inference efficiency.

Algorithm 1 Forward Pass for Lagrangian Dual Layer (LDL)

Input: θ

Output: $x = LDL(\theta)$

Constants: G, B, h, p

Initialization: $\alpha = 0, mt = 0, vt = 0, \mu = 100, b_1 = 0.9, b_2 = 0.99, e = 10^{-9}, decay\_rate = 0.95$

$\lambda = \min(\theta)$

repeat

Stage 1: Calculate original variable

$x = \frac{I \odot \theta - G^\top \alpha}{\lambda}$

$t = \frac{-B^\top \alpha}{\lambda}$

$x = \min(0, \max(x, 1))$

$t = \min(0, \max(t, 1))$

Stage 2: Calculate dual variable gradient

$\frac{\partial L}{\partial \alpha} = h - Gx - Bt$

Stage 3: Update dual variable

$mt = b_1 * mt + (1 - b_1) * \frac{\partial L}{\partial \alpha}$

$vt = b_2 * vt + (1 - b_2) * (\frac{\partial L}{\partial \alpha} \odot \frac{\partial L}{\partial \alpha})$

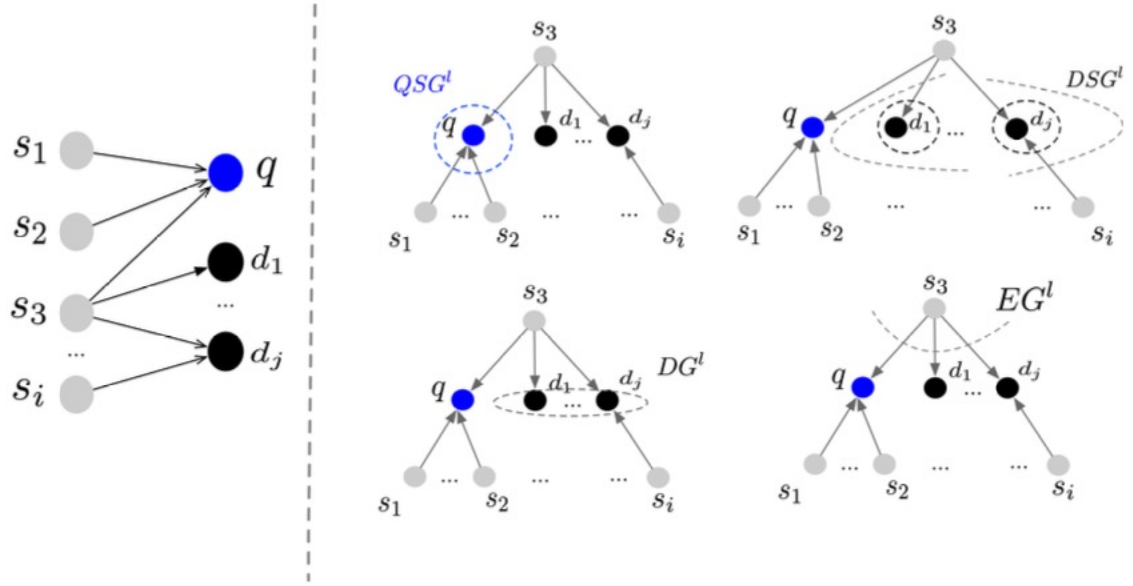
$\alpha = \min(\max(0, \alpha - \mu * \frac{mt}{\sqrt{vt + e}}, p))$

$\mu = \mu * decay\_rate$

until Convergence or maximum iterations

Graph Convolutional Network (GCN) Extracts Meaningful Features

- GCN Module extracts meaningful features from the selling allocation problem as objective and conditions vary greatly between GD contracts
- It captures adaptive features for diverse decision scenarios and improve accuracy and flexibility of allocation optimization



Query to Supply GCN	Demand to Supply GCN	Demand GCN	Edge GCN	Concatenate with ReLU
• Impressions supply is converted to embedding and broadcasted to all supply nodes	• Aggregated supply features from constraint demand nodes	• Aggregate demand node to represent competition for supply impressions	• Edge conditions account for boundary conditions like frequency and crowd	• All features are concatenated into a single layer passed to Multi-layered Perceptron

Forward Computation Completes End-to-End Allocation Optimization

Dense Neural Network (DNN) extracts important features from input bi-graph, Z'

Z' along with GCN Embeddings are inputs to Multi-layered Perceptron (MLP) with ReLU activation to estimate supply impressions $\hat{\theta}$

Estimated supply impressions $\hat{\theta}$ are input to Langrangian dual layer to produce estimated allocation ratio $\hat{x}$

Allocation regret is minimized via custom training loss function

$$\left|(I \odot \hat{\theta})'\hat{x} - (I \odot \theta)'x\right|_2^2 + \epsilon\left|\theta - \hat{\theta}\right|_2^2$$

Training loss has two parts: end-to-end allocation regret and prediction error for inventory nodes. Hyperparameter ϵ balances their importances

Algorithm 2 Forward Pass for End-to-End Allocation Optimization

Input: Z^0, D, C
Output: x

$Constant : G, B, h, p, A, I$

$Z' = DNN(Z^0)$

for $l = 1, \cdots, L$ **do**

$$QSG^l = Broadcast(Relu(\frac{I^T Z^{(l-1)} W^l_{QSG}}{\sum_{i=0}^{m} A_{ij}}))$$

$$DSG^l = Relu(\frac{A}{\sum_{j=0}^{n} A_{ij}} \frac{A^T Z^{(l-1)}}{\sum_{i=0}^{m} A_{ij}} W^l_{DSG})$$

$$DG^l = Relu(\frac{ADW^l_{DG}}{\sum_{j=0}^{n} A_{ij}})$$

$$EG^l = Relu(\frac{\sum_{j=0}^{n}(A_{ij} \cdot C_{ij})}{\sum_{j=0}^{n} A_{ij}} W^l_{EG})$$

$$Z^l = Relu((QSG^l, DSG^l, DG^l, EG^l) \cdot W_Z)$$

end for

$\hat{\theta} = Relu(MLP(Z^L, Z'))$

$x = LDL(\hat{\theta})$

Evaluation

How does our model perform in comparison to existing methods?

We evaluate our model via two daily datasets

Offline Dataset

- The aim is to test the feasibility of end-to-end optimization theory, comparing the two-stage method and end-to-end approach across three inventory allocation cases: full targeting, single targeting, and random targeting
- Basic constraints (overselling and underdelivery) are added to the offline inventory allocation cases
- *Supply x Demand* dimension is *100 x 10* with only 110 constraints (100 overselling and 10 underdelivery)

Online Dataset

- Experiments are conducted on two online advertising selling datasets: Pre-Video Ads (PVA) and Open-Screen Ads (OSA)
- Each supply node of PVA represents a *Channel x City x Position* combination; for OSA is *City x App*
- The inventory allocation and allocation problems are more dynamic and complex compared to the offline datasets
- For *m* supply nodes and *n* demand nodes, there are $(m \times n + m + n)$ constraints

Dimensions	**Offline**			**Online PVA**	**Online OSA**
	full targeting	single targeting	random targeting	real targeting	real targeting
supply $\times$ *demand*	100×10	100×10	100×10	500×20	500×50
constraints	110	110	110	**10520**	**25550**

We compare our end-to-end model with five benchmark models on six metrics

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)
- Implements established LP solvers IntOpt and QPTL for comparison with Lagrangian layer

Evaluation Metrics

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

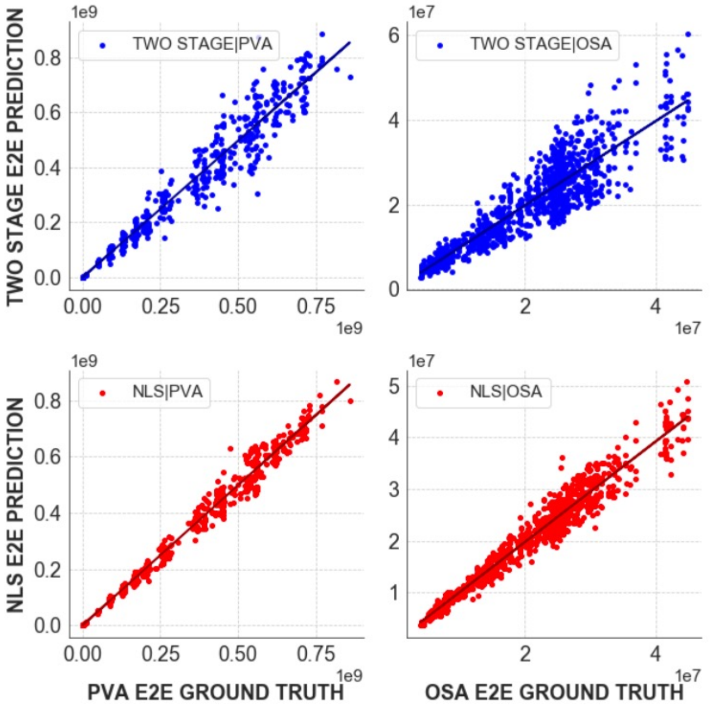
NLS outperforms all other models on most benchmarks

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	$\mathbf{0.096}_{\pm 0.002}$	$\mathbf{0.007}_{\pm 2e-4}$	$\mathbf{0.097}_{\pm 0.001}$	$\mathbf{0.098}_{\pm 0.001}$	$\mathbf{0.095}_{\pm 0.001}$	$\mathbf{0.029}_{\pm 1e-4}$

Table: Experiment Results on Offline 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	$\mathbf{0.064}_{\pm 0.003}$	$\mathbf{0.041}_{\pm 0.001}$	$0.068_{\pm 0.003}$	$\mathbf{0.058}_{\pm 0.001}$

Table: Experiment Results on Online Data



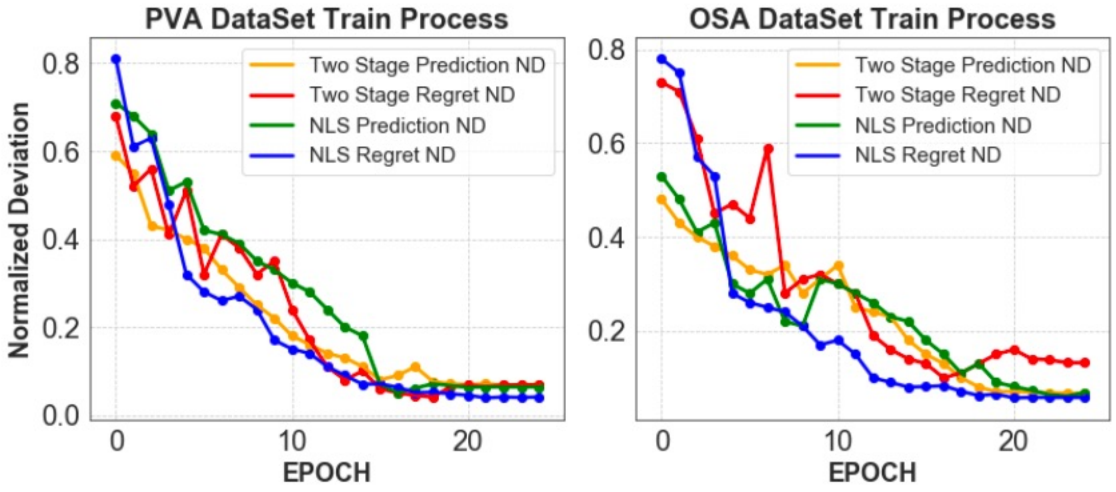
Our NLS has fewer outliers in comparison with two-stage methods.

ND_{pre} and ND_{reg} are Normalized Deviations for prediction and regret.

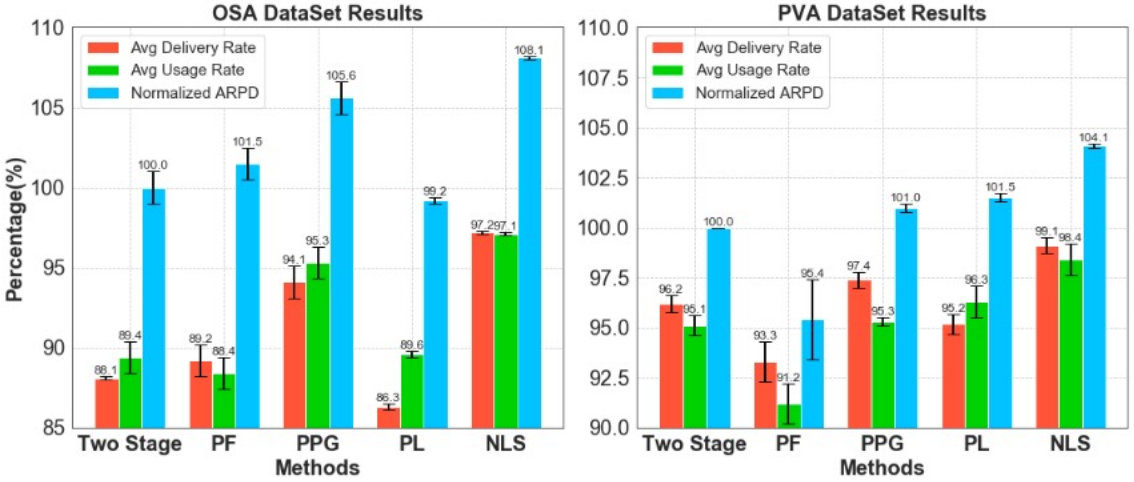
NLS handles complex optimization constraints better than benchmarks

Methods	Offline (random)			PVA			OSA		
	ND_{pre}	ND_{allo}	ND_{reg}	ND_{pre}	ND_{allo}	ND_{reg}	ND_{pre}	ND_{allo}	ND_{reg}
GCN + QPTL	$0.098_{\pm 0.008}$	$0.15_{\pm 0.02}$	$0.032_{\pm 0.005}$	-	-	-	-	-	-
GCN + IntOpt	$0.100_{\pm 0.003}$	$0.25_{\pm 0.03}$	$0.038_{\pm 0.005}$	$0.068_{\pm 0.006}$	$0.32_{\pm 0.04}$	$0.063_{\pm 0.002}$	$0.070_{\pm 0.008}$	$0.33_{\pm 0.04}$	$0.083_{\pm 0.006}$
NLS	$\mathbf{0.095}_{\pm 0.001}$	$\mathbf{1e\text{-}4}_{\pm 1e-5}$	$\mathbf{0.029}_{\pm 1e-4}$	$\mathbf{0.064}_{\pm 0.003}$	$\mathbf{7e\text{-}4}_{\pm 1e-5}$	$\mathbf{0.041}_{\pm 0.0031}$	$\mathbf{0.068}_{\pm 0.003}$	$\mathbf{6e\text{-}4}_{\pm 1e-5}$	$\mathbf{0.058}_{\pm 0.001}$

NLS outperforms classic optimization methods on all benchmarks



NLS has more stable training trajectory



NLS has better delivery rate, usage rate and publisher revenue

ND_{pre} and ND_{allo} are Normalized Deviations for prediction and allocation.
ARPD is Average Revenue per Day.

Conclusion: Key Takeaways and Results

1. **Key Idea:** Proposes Neural Lagrangian Selling (NLS), an end-to-end approach for inventory prediction and contract allocation in Guaranteed Delivery (GD) advertising.
2. **Methodology:** NLS integrates a differentiable Lagrangian dual optimization layer into predictive deep neural networks, overcoming limitations of traditional two-stage methods.
3. **Comparison:** NLS is compared with established models and methods, including Two-Stage, PF, PPG, PL, GCN+QPTL, and GCN+InOpt.
4. **Results:** NLS outperforms existing models in terms of allocation regret minimization and computational efficiency.
5. **Significance:** The end-to-end approach offers significant improvements in the GD selling process by better handling dynamic optimization objectives and constraints.

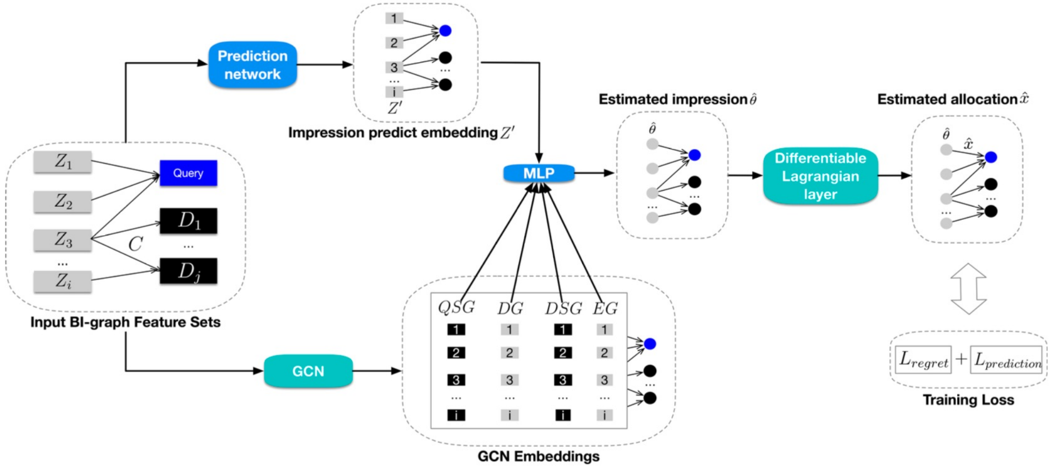


Figure: Neural Langrangian Selling (NLS) Model for End-to-End Prediction and Optimization

Thank you for your time.

Questions?

THE UNIVERSITY OF TENNESSEE KNOXVILLE