Inferring Heterogeneous Private Valuations from Offline Market Data via Entropic Risk-Sensitive Utility Maximization (Supplementary Material)

Xingyu Qian, Haoran Yu*

School of Computer Science & Technology, Beijing Institute of Technology
3120241000@bit.edu.cn, yhrhawk@gmail.com

1 Comparison Baselines

We introduce the details and pseudocode of five baseline methods: **CE**, **BLUE**, **SL**, **DL**, **PGM**, **RvS**.

1.1 CE

This method addresses valuation inference using supervised learning to train a decision function $\pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{v}, \boldsymbol{o}^t)$ that maps valuation $\boldsymbol{v}$ and market observation $\boldsymbol{o}^t$ to buyer behavior probabilities. Since true valuations are unavailable during training, we substitute $\boldsymbol{v}$ with samples from feasible ranges $\mathcal{V}_{\text{feas}}$. The inference phase solves an optimization problem to find valuations $\boldsymbol{v}_n^*$ that maximize the likelihood of observed behaviors.

During the Model Training Phase, we minimize the cross-entropy between predicted and actual behaviors:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\log \pi_{\boldsymbol{\theta}}(b_n^t|\tilde{\boldsymbol{v}}_n, \boldsymbol{o}_n^t). \tag{1}$$

where $\tilde{\boldsymbol{v}}_n$ are feasible valuations sampled from $\mathcal{V}_{n,\text{feas}}$. This trains $\pi_{\boldsymbol{\theta}}$ to predict behaviors given sampled valuations and market observations.

In the Valuation Inference Phase, for fixed $\boldsymbol{\theta}$, we find $\boldsymbol{v}_n^*$ that minimizes the negative log-likelihood of observed behaviors:

$$\boldsymbol{v}_n^* = \arg\min_{\boldsymbol{v}_n \in \mathcal{V}_{n,\text{feas}}} -\sum_{t=0}^{T-1}\log \pi_{\boldsymbol{\theta}}\left(b_n^t|\boldsymbol{v}_n, \boldsymbol{o}_n^t\right). \tag{2}$$

This optimization recovers valuations that best explain observed behaviors under the trained model. The pseudocode is shown in Algorithm 1.

1.2 BLUE

BLUE extends the **CE** method by introducing a rationality constraint during training to ensure economically plausible behavior predictions. This constraint penalizes the decision function when it assigns probability to strictly dominated actions. Specifically, actions should only be plausible for valuations within the buyer's feasible valuation range $\mathcal{V}_{\text{feas}}$.

Algorithm 1: Cross-Entropy Based Valuation Inference

Input: Offline datasets of buyer $\mathcal{D} = \{(\boldsymbol{o}_n^t, b_n^t)\}_{n\in\mathcal{N},t\in\mathcal{T}}$, learning rates $\eta_{\text{train}}, \eta_{\text{inf}}$, the number of training epochs T_{train}, the number of inference iterations T_{inf}.
Output: A set of inferred valuations, i.e., $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

Compute the feasible valuation ranges $\mathcal{V}_{n,\text{feas}}$ for each buyer.
// **Model Training Phase:** Train behavior predictor $\pi_{\boldsymbol{\theta}}$.
Initialize $\boldsymbol{\theta}$ randomly.
for epoch $= 1$ **to** T_{train} **do**
  Shuffle $\mathcal{D}$ and partition into batches $\mathcal{B}$.
  for each batch $\mathcal{B} \subset \mathcal{D}$ **do**
    Sample $\tilde{\boldsymbol{v}}_n \sim \mathcal{V}_{n,\text{feas}}$ for each buyer n.
    Compute $\mathcal{L}_{\text{CE}}$.
    Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_{\text{train}}\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\text{CE}}$.
  end for
end for
// **Valuation Inference Phase:** Recover valuations $\boldsymbol{v}_n^*$.
for each buyer $n \in \mathcal{N}$ **do**
  Initialize $\boldsymbol{v}_n^{(0)}$.
  for iter $= 1$ **to** T_{inf} **do**
    Compute $\mathcal{L} = -\sum_{t=0}^{T-1}\log \pi_{\boldsymbol{\theta}}\left(b_n^t|\boldsymbol{v}_n^{(\text{iter})}, \boldsymbol{o}_n^t\right)$.
    Update $\boldsymbol{v}^{(\text{iter}+1)} \leftarrow \boldsymbol{v}^{(\text{iter})} - \eta_{\text{inf}}\nabla_{\boldsymbol{v}}\mathcal{L}$.
  end for
  $\boldsymbol{v}_n^* \leftarrow \boldsymbol{v}_n^{(T_{\text{inf}})}$.
end for
return $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

The loss function combines standard cross-entropy with a rationality penalty:

$$\mathcal{L}_{\text{BLUE}} = (1-\alpha)\mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{ratio}}, \tag{3}$$

where $\alpha \in [0, 1]$ balances the two objectives. The theoretical formulation represents the negative log probability that valuations belong to the feasible set given observed behaviors:

$$\mathcal{L}_{\text{ratio}} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\log \Pr(\boldsymbol{v}_n \in \mathcal{V}_{n,\text{feas}}|\boldsymbol{o}_n^t, b_n^t). \tag{4}$$

In practical implementation, we approximate this using discrete valuation sampling and probability ratios. The com-

*Corresponding author.

Algorithm 2: Bayesian Learning-based Valuation Inference (BLUE)

Input: Offline datasets of buyer $\mathcal{D} = \{(\boldsymbol{o}_n^t, b_n^t)\}_{n\in\mathcal{N},t\in\mathcal{T}}$, learning rates $\eta_{\text{train}}, \eta_{\text{inf}}$, the number of training epochs T_{train}, the number of inference iterations T_{inf}.
Output: A set of inferred valuations, i.e., $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

Compute the feasible valuation ranges $\mathcal{V}_{n,\text{feas}}$ for each buyer.
Initialize θ randomly.
for $\text{epoch} = 1$ **to** T_{train} **do**
  Shuffle $\mathcal{D}$ and create batches $\{\mathcal{B}\}$.
  for each batch $\mathcal{B}$ **do**
    Sample $\tilde{\boldsymbol{v}}_n \sim \mathcal{V}_{n,\text{feas}}$ for each buyer.
    Compute $\mathcal{L}_{\text{CE}}$ using $\tilde{\boldsymbol{v}}_n$.
    Compute $\mathcal{L}_{\text{ratio}}$ using (5).
    $\mathcal{L}_{\text{total}} = (1-\alpha)\mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{ratio}}$.
    Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_{\text{train}}\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\text{total}}$.
  end for
end for
Infer each buyer's value according to (2).
return $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

putationally efficient version used in code is:

$$\mathcal{L}_{\text{ratio}} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\log\frac{\sum_{v_c\in\mathcal{V}_{n,\text{feas}}}\pi_{\boldsymbol{\theta}}(b_n^t|\boldsymbol{v}_c,\boldsymbol{o}_n^t)}{\sum_{v_c\in\mathcal{V}_{\text{all}}}\pi_{\boldsymbol{\theta}}(b_n^t|\boldsymbol{v}_c,\boldsymbol{o}_n^t)}. \quad (5)$$

Here, $\mathcal{V}_{\text{all}}$ is the complete valuation range considered in the analysis, defined as the discrete set ranging from 20 to 250, containing all possible valuation points.

Algorithm 2 shows the pseudocode. The Valuation Inference Phase in **BLUE** remains identical to the **CE** method.

1.3 SL

SL learns a direct mapping from buyer observations and behaviors to their private valuations. Unlike **CE** and **BLUE** which model the decision function and invert it, **SL** uses a supervised regression approach to predict valuations directly from market observations. The method incorporates critical rationality constraints to ensure predicted valuations are economically plausible.

The loss function aims to maximize the probability that inferred valuations belong to the feasible set $\mathcal{V}_{n,\text{feas}}$ given observed behaviors:

$$\mathcal{L}_{\text{SL}} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\log\Pr(\boldsymbol{v}_n\in\mathcal{V}_{n,\text{feas}}|\boldsymbol{o}_n^t,b_n^t). \quad (6)$$

In code, we implement a computationally efficient approximation using constraint penalties. Let $\hat{\boldsymbol{v}}_n$ denote the valuation predicted by the neural network. Let $\boldsymbol{v}_{n,\min}$ and $\boldsymbol{v}_{n,\max}$ denote the values in the feasible set $\mathcal{V}_{n,\text{feasible}}$ with the lowest L_2-norm and the highest L_2-norm, respectively. The loss function is defined as follows (the operation $\max$ is taken elementwise).

Algorithm 3: Supervised Valuation Inference (SL)

Input: Offline datasets of buyer $\mathcal{D} = \{(\boldsymbol{o}_n^t, b_n^t)\}_{n\in\mathcal{N},t\in\mathcal{T}}$, learning rate η, the number of epochs T_{train},
Output: A set of inferred valuations, i.e., $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

Compute the feasible valuation ranges $\mathcal{V}_{n,\text{feas}}$ for each buyer.
Initialize the neural network $f_{\boldsymbol{\theta}}$.
for $\text{epoch} = 1$ **to** T_{train} **do**
  Shuffle $\mathcal{D}$ and create batches $\{\mathcal{B}\}$.
  for each batch $\mathcal{B}$ **do**
    Predict valuations $\hat{\boldsymbol{v}}_n$ according to $f_{\boldsymbol{\theta}}(\boldsymbol{o}_n^t, b_n^t)$.
    Compute lower penalty $\mathcal{L}_{\text{lower}}$.
    Compute upper penalty $\mathcal{L}_{\text{upper}}$.
    $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{lower}} + \mathcal{L}_{\text{upper}}$.
    Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\text{total}}$.
  end for
end for
Infer each buyer's value with $f_{\boldsymbol{\theta}}$.
return $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

$$\mathcal{L}_{\text{lower}} = \sum_n ||\max(\mathbf{0}, \boldsymbol{v}_{n,\min} - \hat{\boldsymbol{v}}_n)||_2^2. \quad (7)$$

$$\mathcal{L}_{\text{upper}} = \sum_n ||\max(\mathbf{0}, \hat{\boldsymbol{v}}_n - \boldsymbol{v}_{n,\max})||_2^2. \quad (8)$$

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{lower}} + \mathcal{L}_{\text{upper}}. \quad (9)$$

We show the pseudocode in Algorithm 3.

1.4 DL

DL employs a dual learning framework that alternately tackles two interconnected tasks: (i) valuation inference, estimating valuations from observations and behaviors, and (ii) behavior prediction, forecasting behaviors from valuations and observations. This approach creates a mutually constraining relationship where each task provides supervisory signals to the other, enhancing both through iterative refinement.

DL employs two interconnected models with distinct loss functions derived from our baseline methods:

- Behavior Prediction Model ($\pi_{\boldsymbol{\theta}_b}$) uses the cross-entropy loss $\mathcal{L}_{\text{CE}}$ from Equation (1). It maps observations and valuations to behavior probabilities.
- Valuation Inference Model ($f_{\boldsymbol{\theta}_c}$) uses the feasibility constraint loss $\mathcal{L}_{\text{SL}}$ from Equation (6). It maps observations and behaviors to valuations.

In the code implementation, we utilize four specialized loss functions that build upon the theoretical foundations of $\mathcal{L}_{\text{CE}}$ and $\mathcal{L}_{\text{SL}}$, but with specific inputs tailored to each optimization step.

The first loss function is defined as

$$\mathcal{L}_{c1} = \sum_n ||\max(\mathbf{0}, \boldsymbol{v}_{n,\min} - \hat{\boldsymbol{v}}_n^{(1)})||_2^2 + \sum_n ||\max(\mathbf{0}, \hat{\boldsymbol{v}}_n^{(1)} - \boldsymbol{v}_{n,\max})||_2^2, \quad (10)$$

where $\hat{\boldsymbol{v}}_n^{(1)} = f_{\boldsymbol{\theta}_c}(\boldsymbol{o}_n^t, b_n^t)$ is the predicted valuation from the valuation inference model given the true observation and true behavior.

The second loss function is given by

$$\mathcal{L}_{b1} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\log \pi_{\boldsymbol{\theta}_b}(b_n^t|\hat{\boldsymbol{v}}_n^{(1)}, \boldsymbol{o}_n^t). \tag{11}$$

This loss uses the valuation prediction $\hat{\boldsymbol{v}}_n^{(1)}$ from Step 1 as input to the behavior prediction model.

The third loss function is given by

$$\mathcal{L}_{b2} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\log \pi_{\boldsymbol{\theta}_b}(b_n^t|\tilde{\boldsymbol{v}}_n, \boldsymbol{o}_n^t), \tag{12}$$

where $\tilde{\boldsymbol{v}}_n$ is a valuation sampled from the feasible range $\mathcal{V}_{n,\text{feas}}$. The output $\hat{b}_n^t$ is the predicted behavior generated by the behavior model using this sampled valuation.

The fourth loss function is given by

$$\begin{aligned}\mathcal{L}_{c2} = &\sum_n ||\max(\mathbf{0}, \boldsymbol{v}_{n,\min} - \hat{\boldsymbol{v}}_n^{(2)})||_2^2 \\ &+ \sum_n ||\max(\mathbf{0}, \hat{\boldsymbol{v}}_n^{(2)} - \boldsymbol{v}_{n,\max})||_2^2,\end{aligned} \tag{13}$$

where $\hat{\boldsymbol{v}}_n^{(2)} = f_{\boldsymbol{\theta}_c}(\boldsymbol{o}_n^t, \hat{b}_n^t)$ is the valuation predicted using the behavior output $\hat{b}_n^t$ from Step 3 as input to the valuation inference model. Algorithm 4 shows the pseudocode.

1.5 PGM

PGM searches for valuations that minimize the norm of the bidding policy's gradient, motivated by the principle that the optimal bidding policy has a zero gradient when it takes the true valuation as input. This method combines inverse reinforcement learning with gradient-based optimization to recover valuations by analyzing policy stability.

The method exploits the equilibrium condition where the optimal bidding policy π^E satisfies:

$$\nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}^E(\boldsymbol{v}^*, \boldsymbol{o}) = \mathbf{0} \tag{14}$$

for true valuation $\boldsymbol{v}^*$, meaning that the policy gradient vanishes at optimality. **PGM** inverts this relationship to find valuations that minimize the policy gradient norm.

The objective function for valuation recovery is

$$\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}\in\mathcal{V}_{n,\text{feas}}} ||\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{v})||_2^2, \tag{15}$$

where the policy loss $\mathcal{L}$ is computed as

$$\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{v}_n) = -\frac{1}{T}\sum_{t=1}^{T}\log \pi_{\boldsymbol{\theta}}^E(b_n^t|\boldsymbol{v}_n, \boldsymbol{o}_n^t)\cdot G_t, \tag{16}$$

with $G_t = \sum_{k=t}^{T}\gamma^{k-t}r_k$ being the discounted accumulated return. Algorithm 5 shows the pseudocode.

Algorithm 4: Dual Learning Framework

Input: Offline datasets of buyer $\mathcal{D} = \{(\boldsymbol{o}_n^t, b_n^t)\}_{n\in\mathcal{N},t\in\mathcal{T}}$, learning rates η_c and η_b, the number of epochs T_{train}.
Output: A set of inferred valuations, i.e., $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

Compute the feasible valuation ranges $\mathcal{V}_{n,\text{feas}}$ for each buyer.
Initialize valuation inference model $f_{\boldsymbol{\theta}_c}$ and behavior prediction model $\pi_{\boldsymbol{\theta}_b}$.
for epoch $= 1$ **to** T_{train} **do**
  Shuffle $\mathcal{D}$ and create batches $\{\mathcal{B}\}$.
  for each batch $\mathcal{B}$ **do**
    // Step 1: Update valuation model with true behavior.
    $\hat{\boldsymbol{v}}_n^{(1)} \leftarrow f_{\boldsymbol{\theta}_c}(\boldsymbol{o}_n^t, b_n^t)$.
    Compute $\mathcal{L}_{c1}$.
    $\boldsymbol{\theta}_c \leftarrow \boldsymbol{\theta}_c - \eta_c\nabla_{\boldsymbol{\theta}_c}\mathcal{L}_{c1}$.
    // Step 2: Update behavior model with predicted valuations.
    Compute $\mathcal{L}_{b1}$ using $\hat{\boldsymbol{v}}_n^{(1)}$.
    $\boldsymbol{\theta}_b \leftarrow \boldsymbol{\theta}_b - \eta_b\nabla_{\boldsymbol{\theta}_b}\mathcal{L}_{b1}$.
    // Step 3: Update behavior model with sampled valuations.
    Sample $\tilde{\boldsymbol{v}}_n \in \mathcal{V}_{n,\text{feas}}$ for each buyer.
    Compute $\mathcal{L}_{b2}$ using $\tilde{\boldsymbol{v}}_n$.
    $\boldsymbol{\theta}_b \leftarrow \boldsymbol{\theta}_b - \eta_b\nabla_{\boldsymbol{\theta}_b}\mathcal{L}_{b2}$.
    // Step 4: Update valuation model with predicted behavior.
    $\hat{b}_n^t \leftarrow f_{\boldsymbol{\theta}_b}(\tilde{\boldsymbol{v}}_n, \boldsymbol{o}_n^t)$.
    $\hat{v}_n^{(2)} \leftarrow f_{\boldsymbol{\theta}_c}(\boldsymbol{o}_n^t, \hat{b}_n^t)$.
    Compute $\mathcal{L}_{c2}$.
    $\boldsymbol{\theta}_c \leftarrow \boldsymbol{\theta}_c - \eta_c\nabla_{\boldsymbol{\theta}_c}\mathcal{L}_{c2}$.
  end for
end for
Infer each buyer's value with $f_{\boldsymbol{\theta}_c}$.
return $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

1.6 RvS

This method addresses valuation inference using supervised learning to train a decision function $\pi_{\boldsymbol{\theta}}(\cdot|r, \boldsymbol{v}, \boldsymbol{o}^t)$ that maps reward r, valuation $\boldsymbol{v}$, and market observation $\boldsymbol{o}^t$ to buyer behavior probabilities. During training, since true valuations are unavailable, we sample $\boldsymbol{v}$. And compute rewards r based on the current observation $\boldsymbol{o}^t$, the sampled valuation $\tilde{\boldsymbol{v}}$, and the next observation $\boldsymbol{o}^{t+1}$. The inference phase involves sampling rewards from a positive range R_{pos} and then solving an optimization problem to find valuations $\boldsymbol{v}_n^*$ that maximize the likelihood of observed behaviors.

During the Model Training Phase, we minimize the cross-entropy between predicted and actual behaviors:

$$\mathcal{L}_{\text{RvS}} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\log \pi_{\boldsymbol{\theta}}(b_n^t|\tilde{r}_n, \tilde{\boldsymbol{v}}_n, \boldsymbol{o}_n^t). \tag{17}$$

where $\tilde{\boldsymbol{v}}_n$ are sampled and $\tilde{r}_n$ are computed as $\tilde{r}_n = f_r(\boldsymbol{o}_n^t, \tilde{\boldsymbol{v}}_n, \boldsymbol{o}_n^{t+1})$. This trains $\pi_{\boldsymbol{\theta}}$ to predict behaviors given computed rewards, sampled valuations, and market observations.

Algorithm 5: Policy Gradient Minimization (PGM)

Input: Offline datasets of buyer $\mathcal{D} = \{(\boldsymbol{o}_n^t, b_n^t)\}_{n\in\mathcal{N},t\in\mathcal{T}}$, the number of inference iterations T_{inf}, a generative model as the market environment simulator.

Output: A set of inferred valuations, i.e., $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

Use $\mathcal{D}$ to train expert policy $\pi_{\boldsymbol{\theta}}^E$.
for each buyer **do**
  Initialize Bayesian optimizer.
  for $i = 1$ **to** T_{iter} **do**
    Sample valuation candidate $\boldsymbol{v}^{(i)}$ according to the Bayesian optimization method.
    Simulate trajectories τ using $\pi_{\boldsymbol{\theta}}^E(\cdot|\boldsymbol{v}^{(i)}, \cdot)$ and the environment simulator.
    Compute policy loss $\mathcal{L}$ using (16).
    Calculate gradient norm $g^{(i)} = ||\nabla_{\boldsymbol{\theta}}\mathcal{L}||_2^2$.
    Update Bayesian optimizer using the collected values of $(\boldsymbol{v}^{(1)}, g^{(1)}), \ldots, (\boldsymbol{v}^{(i)}, g^{(i)})$.
  end for
  Select $\boldsymbol{v}_n^*$ as the $\boldsymbol{v}^{(i)}$ with the lowest $g^{(i)}$.
end for
return $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

In the Valuation Inference Phase, for fixed $\boldsymbol{\theta}$, we first sample reward r_n from a positive range $\mathcal{R}_{\text{pos}}$, then find $\boldsymbol{v}_n^*$ that minimizes the negative log-likelihood of observed behaviors:

$$\boldsymbol{v}_n^* = \arg\min_{\boldsymbol{v}_n\in\mathcal{V}_{n,\text{feas}}} -\sum_{t=0}^{T-1} \log \pi_{\boldsymbol{\theta}}\left(b_n^t | r_n, \boldsymbol{v}_n, \boldsymbol{o}_n^t\right). \tag{18}$$

This optimization recovers valuations that best explain observed behaviors under the trained model and sampled rewards. The pseudocode is shown in Algorithm 6.

Algorithm 6: RL via Supervised Learning Inference

Input: Offline datasets of buyer $\mathcal{D} = \{(\boldsymbol{o}_n^t, b_n^t)\}_{n\in\mathcal{N},t\in\mathcal{T}}$, learning rates $\eta_{\text{train}}, \eta_{\text{inf}}$, the number of training epochs T_{train}, the number of inference iterations T_{inf}.

Output: A set of inferred valuations, i.e., $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.

// **Model Training Phase:** Train behavior predictor $\pi_{\boldsymbol{\theta}}$.
Initialize $\boldsymbol{\theta}$ randomly.
for epoch $= 1$ **to** T_{train} **do**
  Shuffle $\mathcal{D}$ and partition into batches $\mathcal{B}$.
  for each batch $\mathcal{B} \subset \mathcal{D}$ **do**
    Sample $\tilde{\boldsymbol{v}}_n$ for each buyer n.
    Compute $\tilde{r}_n = f_r(\boldsymbol{o}_n^t, \tilde{\boldsymbol{v}}_n, \boldsymbol{o}_n^{t+1})$ using reward function.
    Compute $\mathcal{L}_{\text{RvS}}$.
    Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_{\text{train}}\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\text{RvS}}$.
  end for
end for
// **Valuation Inference Phase:** Recover valuations $\boldsymbol{v}_n^*$.
for each buyer $n \in \mathcal{N}$ **do**
  Sample reward r_n from a positive range $\mathcal{R}_{\text{pos}}$ and initialize $\boldsymbol{v}_n^{(0)}$.
  for iter $= 1$ **to** T_{inf} **do**
    Compute $\mathcal{L} = -\sum_{t=0}^{T-1} \log \pi_{\boldsymbol{\theta}}\left(b_n^t | \boldsymbol{v}_n^{(\text{iter})}, \boldsymbol{o}_n^t, r_t\right)$.
    Update $\boldsymbol{v}^{(\text{iter}+1)} \leftarrow \boldsymbol{v}^{(\text{iter})} - \eta_{\text{inf}}\nabla_{\boldsymbol{v}}\mathcal{L}$.
  end for
  $\boldsymbol{v}_n^* \leftarrow \boldsymbol{v}_n^{(T_{\text{inf}})}$.
end for
return $\{\boldsymbol{v}_n^*\}_{n\in\mathcal{N}}$.