Linear Upper Confidence Bound Algorithm for Contextual Bandit Problem with Piled Rewards

Kuan-Hao Huang and Hsuan-Tien Lin

Department of Computer Science & Information Engineering
National Taiwan University

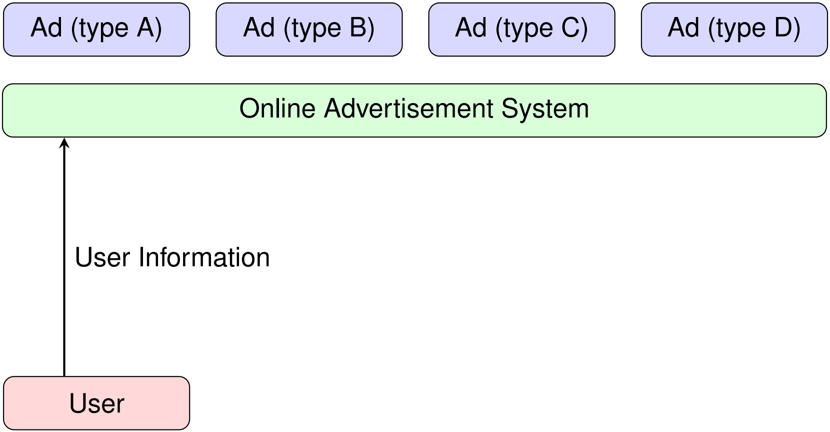
April 20, 2016 (PAKDD)

Contextual Bandit Problem (example)

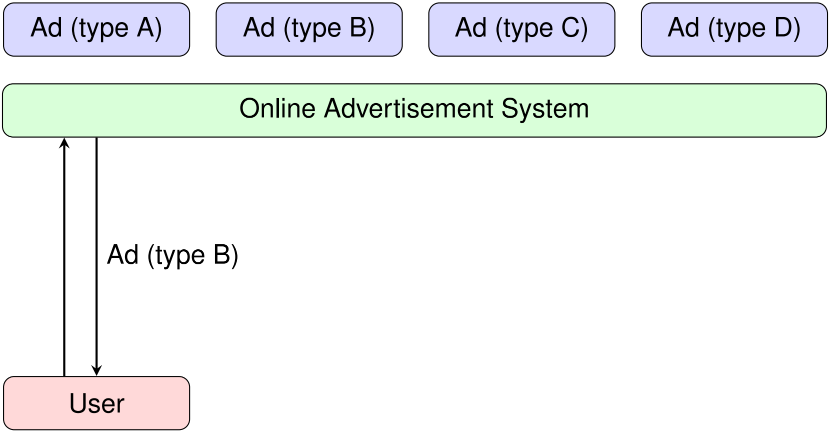
Ad (type A) | Ad (type B) | Ad (type C) | Ad (type D)

Online Advertisement System

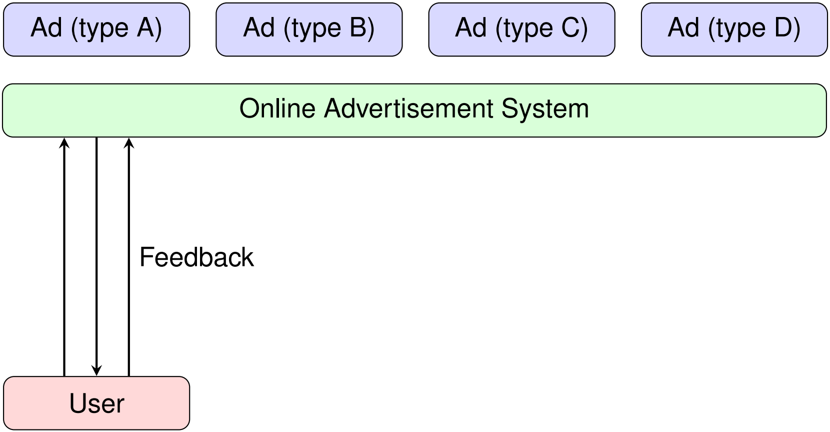
Contextual Bandit Problem (example)



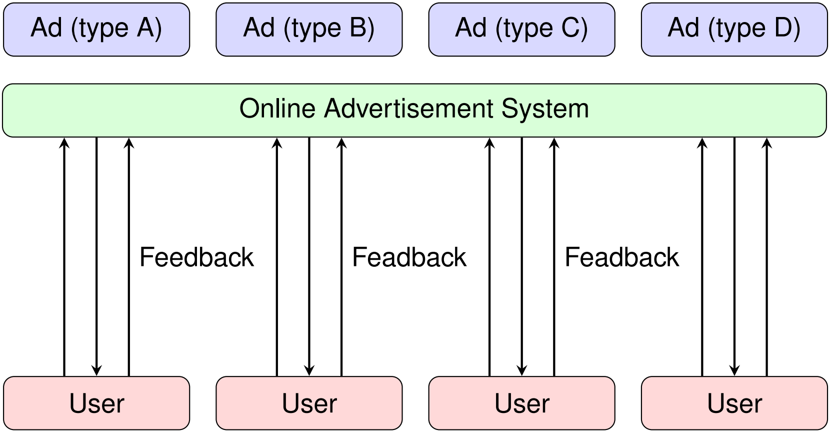
Contextual Bandit Problem (example)



Contextual Bandit Problem (example)



Contextual Bandit Problem (example)



Contextual Bandit Problem (traditional)

Notation

- user: context $\mathbf{x} \in \mathbb{R}^d$
- ad: action $a \in \{1, 2, .., K\}$
- feedback: reward $r \in [0, 1]$

Contextual Bandit Problem (traditional)

Notation

- user: context $\mathbf{x} \in \mathbb{R}^d$
- ad: action $a \in \{1, 2, .., K\}$
- feedback: reward $r \in [0, 1]$

Contextual bandit problem (traditional setting)

for round $t = 1, 2, ..., T$

- algorithm $\mathcal{A}$ receives a context $\mathbf{x}_t$
- algorithm $\mathcal{A}$ selects an action a_t based on the context $\mathbf{x}_t$
- algorithm $\mathcal{A}$ receives the reward r_{t,a_t}

algorithm $\mathcal{A}$ tries to maximize the cumulative rewards $\sum_{t=1}^{T} r_{t,a_t}$

Contextual Bandit Problem (traditional)

Notation

- user: context $\mathbf{x} \in \mathbb{R}^d$
- ad: action $a \in \{1, 2, .., K\}$
- feedback: reward $r \in [0, 1]$

Contextual bandit problem (traditional setting)

for round $t = 1, 2, ..., T$

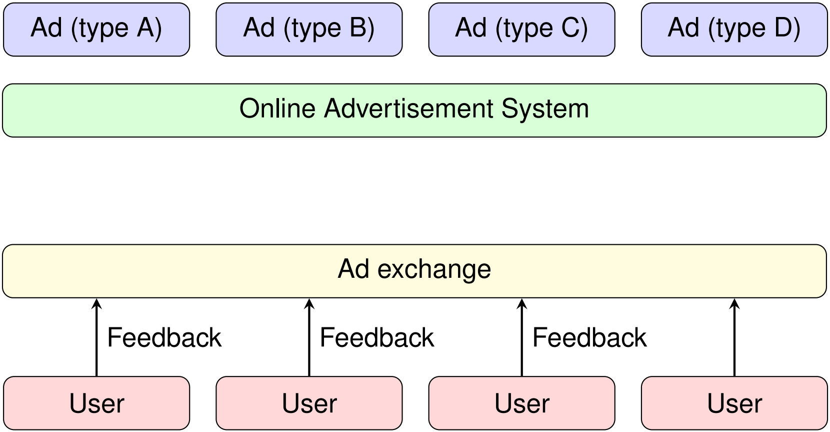
- algorithm $\mathcal{A}$ receives a context $\mathbf{x}_t$
- algorithm $\mathcal{A}$ selects an action a_t based on the context $\mathbf{x}_t$
- algorithm $\mathcal{A}$ receives the reward r_{t,a_t}

algorithm $\mathcal{A}$ tries to maximize the cumulative rewards $\sum_{t=1}^{T} r_{t,a_t}$

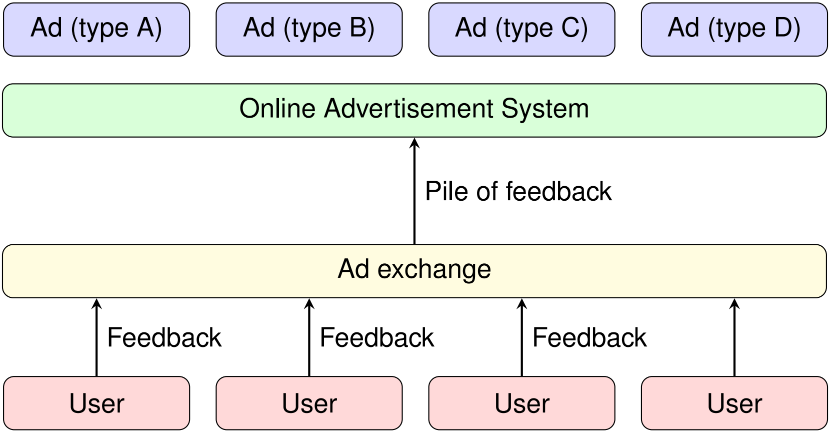
Challenge for contextual bandit problem

- partial feedback: exploitation vs. exploration

Contextual Bandit Problem (piled-reward example)



Contextual Bandit Problem (piled-reward example)



Contextual Bandit Problem (piled-reward)

Notation

- user: context $\mathbf{x} \in \mathbb{R}^d$
- ad: action $a \in \{1, 2, .., K\}$
- feedback: reward $r \in [0, 1]$

Contextual bandit problem (piled-reward setting)

for round $t = 1, 2, ..., T$

- for $i = 1, 2, ..., n$
  - algorithm $\mathcal{A}$ receives a context $\mathbf{x}_{t_i}$
  - algorithm $\mathcal{A}$ selects an action a_{t_i} based on the context $\mathbf{x}_t$
- algorithm $\mathcal{A}$ receives n rewards $r_{t_1,a_{t_1}}, r_{t_2,a_{t_2}}, ..., r_{t_n,a_{t_n}}$

algorithm $\mathcal{A}$ tries to maximize the cumulative rewards $\sum_{t=1}^{T} \sum_{i=1}^{n} r_{t_i,a_{t_i}}$

Linear Upper Confidence Bound (LinUCB)

LinUCB [Li et al., 2010]

- state-of-the-art algorithm for the traditional setting ($n = 1$)
- for each round t and context $\mathbf{x}_t$, LinUCB gives every actions a a score
- selected action $a_t = \text{argmax}_a(\text{score}_{t,a}(\mathbf{x}_t))$

$$
\begin{aligned}
\text{score}_{t,a}(\mathbf{x}_t) &= \text{estimated reward} + \text{uncertainty} \\
&= \text{estimated reward} + \text{confidence bound} \\
&= \mathbf{w}_{t,a}^\top \mathbf{x} + \alpha \sqrt{\mathbf{x}_t^\top (\mathbf{I} + \mathbf{X}_{t-1,a}^\top \mathbf{X}_{t-1,a})^{-1} \mathbf{x}_t}
\end{aligned}
$$

- estimated reward for **exploitation**, is obtained by the online regression from previous pairs $(\mathbf{x}_\tau, r_{\tau,a})$ of action a
- uncertainty for **exploration**, estimates how much confident for the estimated reward
- update the scoring function whenever receiving the reward

Applying LinUCB to Piled-reward setting

LinUCB under the piled-reward setting

for round $t = 1, 2, ..., T$

- for $i = 1, 2, ..., n$
  - LinUCB receives context $\mathbf{x}_{t_i}$
  - LinUCB selects an action a_{t_i} with the same scoring function
- LinUCB receives n rewards $r_{t_1,a_{t_1}}, r_{t_2,a_{t_2}}, ..., r_{t_n,a_{t_n}}$
- LinUCB updates the scoring function with n rewards

Applying LinUCB to Piled-reward setting

LinUCB under the piled-reward setting

for round $t = 1, 2, ..., T$

- for $i = 1, 2, ..., n$
  - LinUCB receives context $\mathbf{x}_{t_i}$
  - LinUCB selects an action a_{t_i} with the same scoring function
- LinUCB receives n rewards $r_{t_1,a_{t_1}}, r_{t_2,a_{t_2}}, ..., r_{t_n,a_{t_n}}$
- LinUCB updates the scoring function with n rewards

Problem for LinUCB under the piled-reward setting

- no update for scoring function within the round
- LinUCB selects action with **high** uncertainty but **low** reward **risk** for some contexts
- these contexts come again and again $\rightarrow$ low cumulative reward
- need **strategic exploration** within the round

Strategic Exploration

Our solution

- use previous contexts $\mathbf{x}_{t_1}, \mathbf{x}_{t_2}, ..., \mathbf{x}_{t_{i-1}}$ in this round to help for selecting the next action for $\mathbf{x}_{t_i}$
- give each previous context $\mathbf{x}_{t_\tau}$ a **pseudo reward** $p_{t_\tau, a_{t_\tau}}$
- use the pseudo reward to **pretend** the true reward
- we design two pseudo rewards:
  - **estimated reward (ER)**: estimated reward
  - **underestimated reward (UR)**: estimated reward - confidence bound

Strategic Exploration

Our solution

- use previous contexts $\mathbf{x}_{t_1}, \mathbf{x}_{t_2}, ..., \mathbf{x}_{t_{i-1}}$ in this round to help for selecting the next action for $\mathbf{x}_{t_i}$
- give each previous context $\mathbf{x}_{t_\tau}$ a **pseudo reward** $p_{t_\tau, a_{t_\tau}}$
- use the pseudo reward to **pretend** the true reward
- we design two pseudo rewards:
  - **estimated reward (ER)**: estimated reward
  - **underestimated reward (UR)**: estimated reward - confidence bound

Score after the update with pseudo reward

pseudo reward	estimated reward	uncertainty
estimated reward	no change	become lower
underestimated reward	become lower	become lower

- achieve **strategic exploration**
- **underestimated reward** is more aggressive than **estimated reward**

Linear Upper Confidence Bound with Pseudo Reward

A novel algorithm

- **Linear Upper Confidence Bound with Pseudo Reward (LinUCBPR)**
  - **LinUCBPR-ER**: estimated reward as the pseudo reward
  - **LinUCBPR-UR**: underestimated reward as the pseudo reward

LinUCBPR under the piled-reward setting

for round $t = 1, 2, ..., T$

- for $i = 1, 2, ..., n$
  - LinUCBPR receives context $\mathbf{x}_{t_i}$
  - LinUCBPR selects an action a_{t_i} with the scoring function
  - LinUCBPR updates the scoring function with the pseudo rewards $p_{t_i, a_{t_i}}$
- LinUCBPR receives n true rewards $r_{t_1, a_{t_1}}, r_{t_2, a_{t_2}}, ..., r_{t_n, a_{t_n}}$
- LinUCBPR discards the change caused by the pseudo rewards
- LinUCBPR updates the scoring function with n true rewards

Theoretical Analysis

Regret of algorithm $\mathcal{A}$

$$\text{Regret}(\mathcal{A}) = \text{perfect reward} - \text{real reward} = \sum_{t=1}^{T}\sum_{i=1}^{n} r_{t_i,a^*_{t_i}} - \sum_{t=1}^{T}\sum_{i=1}^{n} r_{t_i,a_{t_i}}$$

Theorem

With probability $1-\delta$, the regret bounds of LinUCB and LinUCBPR-ER under the piled-reward setting are both

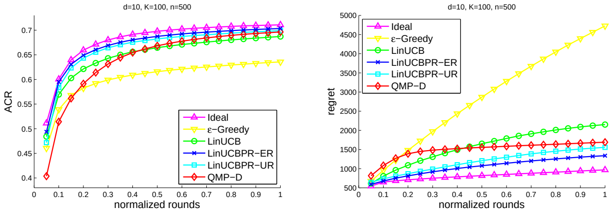
$$\mathcal{O}\left(\sqrt{Cn(nT)\ln^3(nT/\delta)}\right)$$

- when the number of contexts (nT) is constant, the regret bound $\propto \sqrt{n}$
- LinUCB and LinUCBPR-ER enjoy the same regret bound

Artificial Datasets

Artificial data

- $\mathbf{u}_1, \mathbf{u}_1, ..., \mathbf{u}_K \in \mathbb{R}^d$ for K actions
- $r_{t,a} = \mathbf{u}_a^\top \mathbf{x}_t + \epsilon_t$, where $\epsilon \in [-0.05, 0.05]$

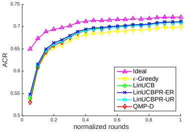


(a) Average cumulative reward (b) Regret

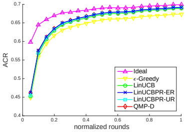
- LinUCBPR outperforms others, especially in the early rounds
- LinUCBPR-ER is better than LinUCBPR-UR

Simple Supervised-to-contextual-bandit Datasets

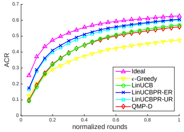
- take supervised-to-contextual-bandit transform [Dudík et al., 2011] on 8 multiclass datasets



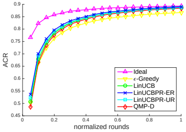
(a) acoustic



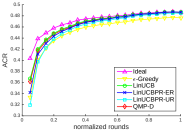
(b) covtype



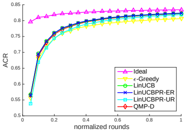
(c) letter



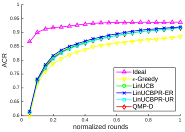
(d) pendigits



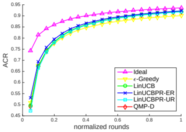
(e) poker



(f) satimage



(g) shuttle



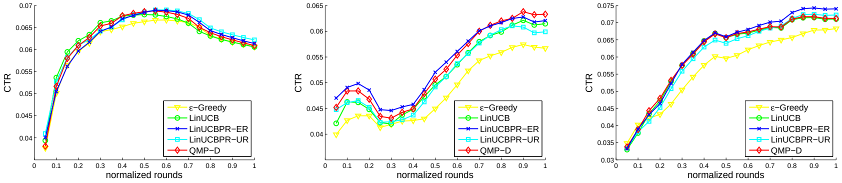
(h) usps

- LinUCBPR-ER reaches the best again

Real-world Datasets

News recommendation dataset by Yahoo!

- appearing in ICML 2012 workshop competition
- the only public dataset for contextual bandit problem
- dynamic action set



(a) R6A (b) R6B (c) R6B-clean

- LinUCBPR-ER is a stable and promising algorithm

Conclusion

- formalize the **piled-reward setting** for contextual bandit problem
- demonstrate how LinUCB can be applied to the piled-reward setting, and prove its **regret bound**
- propose **LinUCBPR**, and prove the **regret bound** of LinUCBPR-ER
- use extensive experiments to validate the **promising performance** of LinUCBPR-ER

Thank you! Any question?