DEVELOPING ADVANCED AI TECHNOLOGIES
FOR BETTER ACCESS OF INFORMATION

HANG LI

AI LAB, BYTEDANCE TECHNOLOGY
WAYS OF INFORMATION ACCESS

Combination of Search, Recommendation, Question Answering, Dialogue

Search  Recommendation  Question Answering  Dialogue
MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE PLAY KEY ROLE
OUTLINE

• *Self-Training Search System: Unbiased LambdaMART*

• *Conversational Recommender System: ConUCB*
SELF TRAINING SEARCH SYSTEM

Autonomous Learning System

\[ f(q, d) \]: ranking model
\[ m(q, d) \]: matching model
\[ g(q) \]: query model
\[ h(d) \]: document model

\{(q, d, c)\}: click data
OPPORTUNITIES AND CHALLENGES

• Opportunities
  • Click Data Usually Represents Users’ Implicit Relevance Feedback
  • Easy to Collect with Low Cost

• Challenges
  • Click Data is Noisy
  • Click Data Has Biases, Including Position Bias, Presentation Bias
  • Click Data May Contain Spam
LEARNING TO RANK

• Learning to Rank = Learning Ranking Model from Data

• Ranking Model: $f(q, d)$ or $f(x)$

• Features: $m(q, d), g(q), h(d), \ldots$

• Training Data: $\{(q, d, r)\}$, Usually Labeled by Humans

• Three Approaches with Different Types of Loss Functions

  • Pointwise Loss Function: $L(f(x_i), r_i^+)$

  • Pairwise Loss Function: $L(f(x_i), f(x_j), r_i^+, r_j^-)$

  • Listwise Loss Function: $L(f(x_1), \ldots, f(x_k), r_1, \ldots, r_k)$
UNBIASED LEARNING TO RANK

• Unbiased Learning to Rank = Learning Ranking Model from Debiased Click Data

• In Self-Training Search System
  • Create Initial Ranker $f^{(0)}$
  • Repeat
    • Collect Click Data $\{(q, d, c)\}$
    • Conduct Debiasing of Click Data $\{(q, d, c) \rightarrow \{(q, d, r)\}\}$
    • Train New Ranker $f^{(i)}$ with Debiased Click Data

• Key Question: How to Eliminate Biases (Position Bias, Presentation Bias)
DEBIASED CLICK DATA AS TRAINING DATA

Query

Click Data

Debiasing

Training

Ranking Model

Implicit Relevance Judgment

Ranking List of Documents
• Eye Tracking Experiment (Joachims et al 2005)
• Results on Top Positions Receive More Attention and More Clicks
• Number of Clicks Decreases from Top to Bottom

Figure 1: Percentage of times an abstract was viewed/clicked depending on the rank of the result.
POINTWISE UNBIASED LEARNING TO RANK (PREVIOUS WORK)

• Pointwise Loss Function (Pointwise Approach)
• Biased: Click = Relevant, Unclick = Irrelevant
• Unbiased (Position Bias): Click = Relevant, Unclick = Irrelevant, with Debiasing
  • Inverse Propensity Weighting Principle
  • Theoretical Guarantee: Unbiased Estimate of Pointwise Relevance Loss
  • Debiasing and Learning Can Be Jointly or Separately Conducted
Conventional Learning to Rank

\[ \int L(f(x_i), r_i^+)dP(x_i, r_i^+) \]

\[ \text{argmin}_f \sum_q \sum_{d_i \in D_q} L(f(x_i), r_i^+) \]

Biased Learning to Rank

\[ \int L(f(x_i), c_i^+)dP(x_i, c_i^+) \]

\[ \text{argmin}_f \sum_q \sum_{d_i \in D_q} L(f(x_i), c_i^+) \]
Bias: Ratio between Click Probability and Relevance Probability

\[ P(c_i^+ | x_i) = t_i^+ P(r_i^+ | x_i) \quad t_i^+ = P(c_i^+ | r_i^+) \]

\[ P(c_i^+ | x_i) = P(c_i^+ | r_i^+) P(r_i^+ | x_i) \text{, if } c^+ \Rightarrow r^+ \]
Bias

\[ P(c_i^+ | x_i) = t_i^+ P(r_i^+ | x_i) \]

**Unbiased Learning to Rank**

\[
\int \frac{L(f(x_i), c_i^+)}{t_i^+} dP(x_i, c_i^+)
\]

\[= \int \frac{L(f(x_i), c_i^+)}{P(c_i^+ | x_i) / P(r_i^+ | x_i)} dP(x_i, c_i^+) \]

\[= \int L(f(x_i), r_i^+) dP(x_i, r_i^+) \]

**Inverse Propensity Weighting**

**Unbiased Estimate**

\[ \text{argmin}_f \sum_q \sum_{d_i \in D_q} \frac{L(f(x_i), c_i^+)}{t_i^+} \]
UNBIASED LEARNING TO RANK

• Wang et al. 2016
  • Employed Pointwise “Inverse Propensity Weighting” Principle, Estimated Position Bias Using Online Randomization

• Joachims et al. 2017
  • Proved Pointwise IPW, Estimated Position Bias Using Online Randomization

• Wang et al. 2018
  • Proposed Method Directly Estimate Position Bias from Click Data, Using Pointwise IPW Principle

• Ai et al. 2018
  • Proposed Joint Learning of Position Bias Model and Ranking Model from Click Data, Again Using Pointwise IPW Principle
PAIRWISE UNBIASED LEARNING TO RANK
(OUR WORK)

• Pairwise Loss Function (Pairwise Approach)

• Biased: Click = Relevant, Unclick = Irrelevant

• Unbiased (Position Bias): Click = Relevant, Unclick = Irrelevant, with Debiasing
  • Inverse Propensity Weighting Principle
  • Theoretical Guarantee: Unbiased Estimate of Pairwise Relevance Loss
  • Debiasing and Learning Can be Jointly or Separately Conducted
Conventional Learning to Rank

\[
\int L(f(x_i), r_i^+, f(x_j), r_j^-) dP(x_i, r_i^+, x_j, r_j^-)
\]

\[
\arg\min_f \sum_q \sum_{(d_i,d_j) \in I_q} L(f(x_i), r_i^+, f(x_j), r_j^-)
\]

Biased Learning to Rank

\[
\int L(f(x_i), c_i^+, f(x_j), c_j^-) dP(x_i, c_i^+, x_j, c_j^-)
\]

\[
\arg\min_f \sum_q \sum_{(d_i,d_j) \in I_q} L(f(x_i), c_i^+, f(x_j), c_j^-)
\]
Biases: Ratio between Click Probability and Relevance Probability,  
Ratio between Unclick Probability and Irrelevance Probability

\[ P(c_i^+ | x_i) = t_i^+ P(r_i^+ | x_i) \quad \text{and} \quad P(c_j^- | x_j) = t_j^- P(r_j^- | x_j) \]

Unbiased Learning to Rank

\[
\int \frac{L(f(x_i), c_i^+, f(x_j), c_j^-))}{t_i^+ \cdot t_j^-} dP(x_i, c_i^+, x_j, c_j^-)
\]

\[
= \int \frac{L(f(x_i), c_i^+, f(x_j), c_j^-))}{P(c_i^+ | x_i)/P(r_i^+ | x_i) P(c_j^- | x_j)/P(r_j^- | x_j)} dP(x_i, c_i^+, x_j, c_j^-)
\]

\[
= \int L(f(x_i), r_i^+, f(x_j), r_j^-)) dP(x_i, r_i^+, x_j, r_j^-)
\]

Inverse Propensity Weighting

Unbiased Estimate

\[
\arg\min_f \sum_q \sum_{(d_i, d_j) \in I_q} \frac{L(f(x_i), c_i^+, f(x_j), c_j^-))}{t_i^+ \cdot t_j^-}
\]
PAIRWISE DEBIASING

\[
\min_{f, t^+, t^-} \mathcal{L}(f, t^+, t^-) = \min_{f, t^+, t^-} \sum_q \sum_{(d_i, d_j) \in I_q} \frac{L(f(x_i), c_i^+, f(x_j), c_j^-)}{t_i^+ t_j^-} + \| t^+ \|_p + \| t^- \|_p
\]

s.t. \( t_1^+ = 1, t_1^- = 1 \)

• Initialize Biases

• Repeat

  • Fix Biases \( t_i^+ \) and \( t_j^- \), Estimate Ranking Model \( f \)
  • Fix Ranking Model \( f \), Estimate Biases \( t_i^+ \) and \( t_j^- \)
Algorithm 1 Unbiased LambdaMART

Require: click dataset \( \mathcal{D} = \{(q, D_q, C_q)\} \); hyper-parameters \( p, M \);
Ensure: unbiased ranker \( f \); propensities \( t^+ \) and \( t^- \);

1: Initialize all propensities as 1;
2: for \( m = 1 \) to \( M \) do
3: for each query \( q \) and each document \( d_i \) in \( D_q \) do
4: Calculate \( \tilde{\lambda}_i \) with \( (t^+)^* \) and \( (t^-)^* \) using (36) and (37);
5: end for
6: Re-train ranker \( f \) with \( \tilde{\lambda} \) using LambdaMART algorithm
7: Re-estimate propensities \( t^+ \) and \( t^- \) with ranker \( f^* \) using (31)
   and (32)
8: end for
9: return \( f, t^+, \) and \( t^- \);
## EXPERIMENTAL RESULTS

Unbiased LambdaMART Significantly Outperforms Existing Methods

<table>
<thead>
<tr>
<th>Ranker</th>
<th>Debiasing Method</th>
<th>MAP</th>
<th>NDCG@1</th>
<th>NDCG@3</th>
<th>NDCG@5</th>
<th>NDCG@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>LambdaMART</td>
<td>Labeled Data (Upper Bound)</td>
<td>0.854</td>
<td>0.745</td>
<td>0.745</td>
<td>0.757</td>
<td>0.790</td>
</tr>
<tr>
<td></td>
<td><strong>Pairwise Debiasing</strong></td>
<td><strong>0.836</strong></td>
<td><strong>0.717</strong></td>
<td><strong>0.716</strong></td>
<td><strong>0.728</strong></td>
<td><strong>0.764</strong></td>
</tr>
<tr>
<td></td>
<td>Regression-EM [22]</td>
<td>0.830</td>
<td>0.685</td>
<td>0.684</td>
<td>0.700</td>
<td>0.743</td>
</tr>
<tr>
<td></td>
<td>Randomization</td>
<td>0.827</td>
<td>0.669</td>
<td>0.678</td>
<td>0.690</td>
<td>0.728</td>
</tr>
<tr>
<td></td>
<td>Click Data (Lower Bound)</td>
<td>0.820</td>
<td>0.658</td>
<td>0.669</td>
<td>0.672</td>
<td>0.716</td>
</tr>
<tr>
<td>DNN</td>
<td>Labeled Data (Upper Bound)</td>
<td>0.831</td>
<td>0.677</td>
<td>0.685</td>
<td>0.705</td>
<td>0.737</td>
</tr>
<tr>
<td></td>
<td>Dual Learning Algorithm [1]</td>
<td>0.828</td>
<td>0.674</td>
<td>0.683</td>
<td>0.697</td>
<td>0.734</td>
</tr>
<tr>
<td></td>
<td>Regression-EM</td>
<td>0.829</td>
<td>0.676</td>
<td>0.684</td>
<td>0.699</td>
<td>0.736</td>
</tr>
<tr>
<td></td>
<td>Randomization</td>
<td>0.825</td>
<td>0.673</td>
<td>0.679</td>
<td>0.693</td>
<td>0.732</td>
</tr>
<tr>
<td></td>
<td>Click Data (Lower Bound)</td>
<td>0.819</td>
<td>0.637</td>
<td>0.651</td>
<td>0.667</td>
<td>0.711</td>
</tr>
<tr>
<td>RankSVM</td>
<td>Labeled Data (Upper Bound)</td>
<td>0.815</td>
<td>0.631</td>
<td>0.649</td>
<td>0.675</td>
<td>0.707</td>
</tr>
<tr>
<td></td>
<td>Regression-EM</td>
<td>0.815</td>
<td>0.629</td>
<td>0.648</td>
<td>0.674</td>
<td>0.705</td>
</tr>
<tr>
<td></td>
<td>Randomization [14]</td>
<td>0.810</td>
<td>0.628</td>
<td>0.644</td>
<td>0.672</td>
<td>0.707</td>
</tr>
<tr>
<td></td>
<td>Click Data (Lower Bound)</td>
<td>0.811</td>
<td>0.614</td>
<td>0.629</td>
<td>0.658</td>
<td>0.697</td>
</tr>
</tbody>
</table>
EXPERIMENTAL RESULTS

• At Commercial Search Engine

• AB Testing: Unbiased LambdaMART vs. LambdaMART + Click Data

• Increasing Click Ratio at Positions 1, 3, 5 by 2.64%, 1.21%, 0.80%
OUTLINE

• Self-Training Search System: Unbiased LambdaMART

• Conversational Recommender System: ConUCB
CONVERSATIONAL RECOMMENDATION

- Better Recommendation through Conversation
- Enhance User Experiences

User

Question

Answer

Item

Rating

Conversation Engine

Recommendation Engine
OPPORTUNITIES AND CHALLENGES

• Opportunities
  • Can Understand User’s Interest Better through Conversation

• Challenges
  • How to Conduct Effective and Efficient Conversation
  • How to Incorporate Result of Conversation into Recommendation
CONVERSATIONAL CONTEXTUAL BANDIT

- Recommendation: Contextual Bandit
- Conversation: Occasionally Asks Questions and Gets Answers
- Goal: to Improve Learning Speed of Contextual Bandit
- Example:
  - Q: Are You Interested in Basketball?
  - A: Yes
OUR APPROACH: CONUCB ALGORITHM

- Extension of UCB Algorithm
- Recommendation Engine: UCB Algorithm
- Conversation Engine: Conversation on Key-Terms

User

Keyterm
Answer
Arm
Reward

Conversation Engine

Recommendation Engine
• Given Context Vector $x_{t,a}$ for Each Arm $a$

• Estimate Reward $r_{t,a}$ for Each Context $x_{t,a}$ based on History

• Select Arm $a_t$

• Receive Reward $r_{t,a_t}$

• News Recommendation

• Arm = Article, Reward = Click

• Context = User + Article
CONTEXTUAL BANDIT

• Estimate Reward from History

\[ r_{t,a} = f(x_a, \theta) \]

• Goal is to Minimize Regret

\[ R(T) = E\left[ \sum_{t=1}^{T} r_{t,a_t^*} \right] - E\left[ \sum_{t=1}^{T} r_{t,a_t} \right] \]

• Trade-off between Exploitation and Exploration
UCB ALGORITHM

• Given Context Vector $x_{t,a}$ for Each Arm $a$

• Estimate Reward $r_{t,a}$ for Each Context $x_{t,a}$ based on History

• Estimate Confidence Interval $c_{t,a}$

• Select arm $a_t = \arg\max_{a \in A_t} r_{t,a} + c_{t,a}$

• Receive Reward $r_{t,a_t}$
LINUCB ALGORITHM

• Given Context Vector $x_{t,a}$ for Each Arm $a$

• Estimate Reward $r_{t,a}$ for Each Context $x_{t,a}$ based on History

• Reward Function with Parameter $\theta$ is Defined as

$$r_{t,a} = \theta^T x_{t,a} + \epsilon_t$$

• Estimate Confidence Interval $c_{t,a}$

• Select Arm $a_t = \arg\max_{a \in A_t} r_{t,a} + c_{t,a}$

• Receive Reward $r_{t,a_t}$

$$\theta_t = \arg\min_\theta \sum_{\tau=1}^{t-1} (\theta^T x_{\tau,a_\tau} - r_{\tau,a_\tau})^2 + \lambda ||\theta||^2_2$$
KEY-TERM AND ARTICLE BIPARTITE GRAPH

• News Recommendation
• Item = Article, Reward = Click
• Question = Key Term, Answer = Yes/No
• Bipartite Graph of Key Terms and Articles
• Key Terms Represent Topics, Entities, etc
• Edges Represent Association
• Weights on Edges Represent Strengths of Association
CONVERSATION FREQUENCY

- Function of Conversation Frequency
  - $b(t) = t$, Converse at Every Round
  - $b(t) = 0$, Does Not Converse
  - $b(t) = \left\lfloor \frac{t}{m} \right\rfloor$, Converse at Every $m$ Rounds
  - $b(t) = \left\lfloor \log \frac{t}{m} \right\rfloor$, Converse Gradually Less Frequently

- Indicator Function of Conversation
  - $q(t) = b(t) - b(t - 1)$
PROBLEM SETTING OF CONUCB

User: \( u \)

Keyterm: \( k \)

Answer: \( \tilde{r} \)

Arm: \( a \)

Reward: \( r \)

Context: \( x_a \)

System

History

\( a_1, r_1, a_1 \)
\( a_2, r_2, a_2 \)
\( \ldots \)
\( a_t, r_t, a_t \)
\( \ldots \)
\( a_T, r_T, a_T \)

\( k_1, \tilde{r}_1, k_1 \)
\( k_t, \tilde{r}_t, k_t \)
\( \ldots \)
CONUCB ALGORITHM

• Given Context Vector $x_{t,a}$ for Each Arm $a$

• If $q(t) = 1$
  • Select Key-Term $k_t$
  • Receive Answer $\tilde{r}_{t,k_t}$

• Estimate Reward $r_{t,a}$ for Each Context $x_{t,a}$ based on History in Both Conversational and Behavioral Feedbacks

• Estimate Confidence Interval $c_{t,a}$

• Select Arm $a_t = \text{argmax}_{a \in A_t} r_{t,a} + c_{t,a}$

• Receive Reward $r_{t,a_t}$
KEY IDEAS OF CONUCB

• Feedbacks on Arms Can Be Propagated to Feedbacks on Key-Terms through Bipartite Graph

\[
E[\tilde{r}_{t,k}] = \sum_{a \in \mathcal{A}} \frac{w_{a,k}}{\sum_{a' \in \mathcal{A}} w_{a',k}} E[r_{t,a}]
\]

• Estimate Parameter \( \tilde{\theta}_t \) based on Feedbacks on Key-Terms at Round \( t \)

\[
\tilde{\theta}_t = \arg \min_{\tilde{\theta}} \sum_{t=1}^{T} \sum_{k \in \mathcal{K}_t} \left( \frac{\sum_{a' \in \mathcal{A}} w_{a',k} \tilde{\theta}^T x_{t,a'}}{\sum_{a' \in \mathcal{A}} w_{a',k}} - \tilde{r}_{t,k} \right)^2 + \lambda \|	ilde{\theta}\|_2^2
\]
KEY IDEAS OF CONUCB

• Estimate Parameter $\theta$ based on Arms at Round $t$

\[
\theta_t = \arg\min_{\theta} \lambda \sum_{\tau=1}^{t-1} (\theta^T x_{\tau,a_{\tau}} - r_{\tau,a_{\tau}})^2 + (1 - \lambda) ||\theta - \tilde{\theta}_t||_2^2
\]

• Estimated Parameter to Conduct Regularization

• There Are Closed Form Solutions for Estimation of $\theta_t$ and $\tilde{\theta}_t$

• Confidence Interval $c_{t,a}$ Can Also Be Estimated

• Arm Selection

\[
a_t = \arg\max_{a \in \mathcal{A}_t} (x_{t,a}^T \theta_t + c_{t,a})
\]
KEY IDEAS OF CONUCB

- Minimize Upper Bound to Select Best Key Term
  \[ E[||X_t \theta_t - X_t \theta_*||^2] \leq \text{upper bound} \]
- Upper Bound Is Function of Information Until Round \( t \)
- Closed Form Solution for Key Term Selection
Algorithm 2: ConUCB algorithm

**Input:** graph \((\mathcal{A}, \mathcal{K}, W)\), conversation frequency function 
\(b(t)\).

**Init:** \(\tilde{M}_0 = \tilde{\lambda}I, \tilde{b}_0 = 0, M_0 = (1 - \lambda)I, b_0 = 0\).

1. for \(t = 1, 2, \ldots, T\) do
2.     if \(b(t) - b(t - 1) > 0\) then
3.         \(\text{nq}= b(t) - b(t - 1)\);
4.         while \(\text{nq}\geq 0\) do
5.             Select a key-term \(k \in \mathcal{K}\) according to Eq. (8), and query the user’s preference over it;
6.             Receive the user’s feedback \(\tilde{r}_{k,t}\);
7.             \(\tilde{M}_t = \tilde{M}_{t-1} + \left( \frac{\sum_{a \in \mathcal{A}} w_{a,k} x_{a,t}}{\sum_{a \in \mathcal{A}} w_{a,k}} \right) \left( \frac{\sum_{a \in \mathcal{A}} w_{a,k} x_{a,t}}{\sum_{a \in \mathcal{A}} w_{a,k}} \right)^T\);
8.             \(\tilde{b}_t = \tilde{b}_{t-1} + \left( \frac{\sum_{a \in \mathcal{A}} w_{a,k} x_{a,t}}{\sum_{a \in \mathcal{A}} w_{a,k}} \right) \tilde{r}_{k,t}\);
9.             nq = 1
10.     else
11.         \(\tilde{M}_t = \tilde{M}_{t-1}, \tilde{b}_t = \tilde{b}_{t-1}\);
12.     \(\tilde{\theta}_t = \tilde{M}_t^{-1} \tilde{b}_t, \theta_t = M_t^{-1} \left( b_t + (1 - \lambda)\tilde{\theta}_t \right)\);
13.     Select \(a_t = \arg \max_{a \in \mathcal{A}} x_{a,t}^T \theta_t + \lambda \alpha_t \| x_{a,t} \|_{M_t^{-1}} + (1 - \lambda)\tilde{\alpha}_t \| x_{a,t}^T M_t^{-1} \|_{\tilde{M}_t^{-1}}\);
14.     Ask the user’s preference on arm \(a_t \in \mathcal{A}\) and receive the reward \(r_{a_t,t}\);
15.     \(M_t = M_t + \lambda x_{a_t,t} x_{a_t,t}^T, b_t = b_t + \lambda x_{a_t,t} r_{a_t,t}\);
THEORETICAL RESULT

• Regret Upper Bound of LinUCB

\[ O(\sqrt{d \log T}) = O((1 - \sqrt{\lambda})\sqrt{d \log T} + \sqrt{\lambda d \log T}) \]

• Regret Upper Bound of ConUCB

\[ O((1 - \sqrt{\lambda})\sqrt{d + \log T} + \sqrt{\lambda d \log T}) \]

• \( T \) is Total Number of Rounds, \( d \) is Dimensionality of Parameter \( \theta \), \( \lambda \in [0, 0.5] \) is Hyperparameter

• ConUCB Has Faster Learning Rate
EXPERIMENTAL RESULTS

ConUCB Outperforms Baselines Like LinUCB on Synthetic, Yelp, and Toutiao Datasets
SUMMARY

• Way of Information Access Is Still Evolving
• Machine Learning Can Play Big Role
• Self-Training Search System
• We Propose Unbiased LambdaMART
• Conversational Recommender System
• We Propose ConUCB
TAKE-AWAY MESSAGE

• Bias in Data Can Be Eliminated in Machine Learning Method
• Conversation Can Be Incorporated into Reinforcement Learning Model
PAPERS AND CODES

• Ziniu Hu, Yang Wang, Qu Peng, Hang Li, Unbiased LambdaMART: An Unbiased Pairwise Learning-to-Rank Algorithm, WebConf 2019, Codes at GitHub

• Xiaoying Zhang, Hong Xie, Hang Li, John Lui, Conversational Contextual Bandits: Algorithm and Application, in Submission, 2019
WE ARE HIRING!

lihang.lh@bytedance.com