Computationally Efficient Optimization of Plackett-Luce Ranking Models for Relevance and Fairness

Harrie Oosterhuis
July 12, 2021

Radboud University
harrie.oosterhuis@ru.nl
https://twitter.com/HarrieOos
Goal of this work:

- Optimize a **Plackett-Luce** (PL) model for **relevance** or **fairness** ranking metrics,
- with an unbiased method (no heuristic or bounding),
- in a **computationally efficient** way (avoid combinatorial problems).

Contribution: **PL-Rank**

- A novel **sampling-based** method for quickly estimating PL gradients.
- Derivation to prove the estimation is unbiased.
Motivation
Traditionally a ranking \textbf{model} $m$ tries to \textbf{score} items $d \in D$ in \textbf{order of relevance}:

$$d \succ_{\text{relevance}} d' \rightarrow m(d) > m(d').$$
Traditionally a ranking model $m$ tries to score items $d \in D$ in order of relevance:

$$d \succ_{\text{relevance}} d' \rightarrow m(d) > m(d').$$

In recent years, probabilistic ranking models have been argued for:

**Fairness** (Singh and Joachims, 2019; Diaz et al., 2020)

- a deterministic ranking will give most attention to a single item, even if there are (almost) equally relevant items.
- A stochastic ranking model can more fairly distribute exposure over items.
Motivation: Stochastic Ranking

Traditionally a ranking model $m$ tries to score items $d \in D$ in order of relevance:

$$d \succ_{\text{relevance}} d' \rightarrow m(d) > m(d').$$

In recent years, probabilistic ranking models have been argued for:

**Fairness** (Singh and Joachims, 2019; Diaz et al., 2020)

- a deterministic ranking will give most attention to a single item, even if there are (almost) equally relevant items.
- A stochastic ranking model can more fairly distribute exposure over items.

**Exploration** (Hofmann et al., 2011; Oosterhuis and de Rijke, 2021)

- when learning from user clicks, a stochastic ranking model can try various rankings according to its uncertainty.
For any ranking $y$, an arbitrary ranking metric uses the weights per rank $\theta_k$, the relevance of the items $P(R = 1 \mid q, d) = \rho_d$, and the policy $\pi$ with the probability of a ranking $\pi(y \mid q)$:

$$
\mathcal{R}(q) = \sum_{y \in \pi} \pi(y \mid q) \sum_{k=1}^{K} \theta_k P(R = 1 \mid q, y_k) = \sum_{y \in \pi} \pi(y) \sum_{k=1}^{K} \theta_k \rho_{y_k} = \mathbb{E}_y \left[ \sum_{k=1}^{K} \theta_k \rho_{y_k} \right].
$$

This is taken in expectation over a query distribution:

$$
\mathcal{R} = \mathbb{E}_q[\mathcal{R}(q)] = \sum_{q \in \mathcal{Q}} P(q) \mathcal{R}(q).
$$

This description applies to well-known metrics: precision@k, recall@k, DCG, ARP.
Background: Plackett-Luce Models
A Plackett-Luce model (Plackett, 1975; Luce, 2012) assumes the probability of selecting an item \( d \) is determined by the value of it compared to the sum of values over all items:

\[
P(d \mid D) = \frac{\text{value of item } d}{\sum_{d' \in D} \text{value of item } d'}.
\]
A Plackett-Luce model (Plackett, 1975; Luce, 2012) assumes the **probability of selecting** an item $d$ is determined by the value of it compared to the **sum of values** over all items:

$$P(d \mid D) = \frac{\text{value of item } d}{\sum_{d' \in D} \text{value of item } d'}.$$ 

A **SoftMax** function is an instance of a Plackett-Luce model, where the exponential function ensures **positive non-zero** values:

$$P(d \mid D) = \frac{e^{m(d)}}{\sum_{d' \in D} e^{m(d')}}.$$
A Plackett-Luce ranking model is repeatedly applied to the *unplaced items*:

\[
\pi(d \mid y_{1:k}, D) = \frac{1[d \notin y_{1:k}]e^{m(d)}}{\sum_{d' \in D \setminus y_{1:k}} e^{m(d')}}.
\]

item score if not placed

sum of all unplaced item scores
A Plackett-Luce ranking model is repeatedly applied to the *unplaced items*:

\[
\pi(d \mid y_{1:k}, D) = \frac{\mathbf{1}[d \notin y_{1:k}] e^{m(d)}}{\sum_{d' \in D \setminus y_{1:k}} e^{m(d')}}.
\]

The probability of a ranking is the product over each item placement:

\[
\pi(y) = \prod_{k=1}^{K} \pi(y_k \mid y_{1:k-1}, D).
\]

We can sample from a Plackett-Luce ranking model by sampling Gumbel Noise: \( \zeta_d \sim Gumbel \), and sorting according to \( m(d) + \zeta_d \) (Bruch et al., 2020).
Background: Policy Gradients

The prevalent approach in existing work (Singh and Joachims, 2019; Bruch et al., 2020) uses **policy-gradients** with the log-trick (Williams, 1992):

\[
\frac{\delta}{\delta m} \pi(y) = \pi(y) \left[ \frac{\delta}{\delta m} \log(\pi(y)) \right].
\]

Given \(N\) samples from \(\pi: y^{(i)} \sim \pi\), the gradient can be **unbiasedly** estimated:

\[
\frac{\delta}{\delta m} R(q) \approx \frac{1}{N} \sum_{i=1}^{N} \left[ \frac{\delta}{\delta m} \log(\pi(y^{(i)})) \right] \left( \sum_{k=1}^{K} \theta_k \rho_{y^{(i)}_k} \right).
\]

- gradient w.r.t. log prob. of full ranking
- observed reward
Method: PL-Rank
A reward before rank $k$ should not influence the probabilities of the ranking after $k$: 

$$
R(q) = \sum_{y \in \pi} \pi(y) \sum_{k=1}^{K} \theta_k \rho_{y_k} = \sum_{k=1}^{K} \theta_k \sum_{y \in \pi} \pi(y) \rho_{y_k} = \sum_{k=1}^{K} \theta_k \sum_{y_{1:k} \in \pi} \pi(y_{1:k}) \rho_{y_k}.
$$
Method: Placement Policy Gradients

A reward before rank \( k \) should not influence the probabilities of the ranking after \( k \):

\[
R(q) = \sum_{y \in \pi} \pi(y) \left( \sum_{k=1}^{K} \theta_k \rho_{y_k} \right) = \sum_{k=1}^{K} \theta_k \sum_{y \in \pi} \pi(y) \rho_{y_k} = \sum_{k=1}^{K} \theta_k \sum_{y_{1:k} \in \pi} \pi(y_{1:k}) \rho_{y_k}.
\]

Given \( N \) samples from \( \pi \): \( y^{(i)} \sim \pi \), the gradient can be unbiasedly estimated:

\[
\frac{\delta}{\delta m} R(q) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \left[ \frac{\delta}{\delta m} \log(\pi(y_k^{(i)} | y_{1:k-1}^{(i)})) \right] \sum_{x=k}^{K} \theta_x \rho_{y_x^{(i)}}.
\]

log prob. of item placement at \( k \) \( \quad \) reward received after \( k \)
Using the fact that $\pi$ is a Plackett-Luce model, we can estimate the gradient using:

$$\frac{\delta}{\delta m} R(q) \approx \sum_{d \in D} \left[ \frac{\delta}{\delta m} m(d) \right] \frac{1}{N} \sum_{i=1}^{N} \left( \sum_{k=\text{rank}(d,y^{(i)})}^{K} \theta_k \rho_{y_k^{(i)}} \right) \left( \sum_{x=1}^{K} \theta_x \rho_{y_x^{(i)}} \right) \cdot \pi(d \mid y_{1:k-1}^{(i)}) \pi(d \mid y_{1:k}^{(i)}) \left( \sum_{k=1}^{K} \theta_x \rho_{y_x^{(i)}} \right).$$

Given $N$ samples, this can be computed in $O(N \cdot K \cdot D)$. 

Flaw: items that are not in the top-$K$ of any of the $N$ sampled rankings will always have a negative gradient.
Method: PL-Rank-1

Using the fact that $\pi$ is a Plackett-Luce model, we can estimate the gradient using:

$$
\frac{\delta}{\delta m} \mathcal{R}(q) \approx \sum_{d \in D} \left[ \frac{\delta}{\delta m} m(d) \right] \frac{1}{N} \sum_{i=1}^{N} \left( \sum_{k=\text{rank}(d,y^{(i)})}^{K} \theta_k \rho_y y_k^{(i)} \right) - \sum_{k=1}^{\text{rank}(d,y^{(i)})} \pi(d \mid y_1^{(i)} \cdots y_{k-1}^{(i)}) \left( \sum_{x=k}^{K} \theta_x \rho_y y_x^{(i)} \right).
$$

Given $N$ samples, this can be computed in $O(N \cdot K \cdot D)$.

**Flaw:** items that are not in the top-$K$ of any of the $N$ sampled rankings will always have a **negative** gradient.
We can avoid the flaw while maintaining the $O(N \cdot K \cdot D)$ complexity:

$$\frac{\delta}{\delta m} R(q) \approx \sum_{d \in D} \left[ \frac{\delta}{\delta m} m(d) \right] \left( 1 \sum_{i=1}^{N} \frac{K}{N} \sum_{k=\text{rank}(d,y^{(i)})+1}^{K} \theta_k \rho_{y^{(i)}_k} \right)$$

$$+ \sum_{k=1}^{\text{rank}(d,y^{(i)})} \pi(d | y^{(i)}_{1:k-1}) \left( \theta_k \rho_d - \sum_{x=k}^{K} \theta_x \rho_{y^{(i)}_x} \right).$$

grad. w.r.t. score

future reward after placement

expected direct reward minus the risk of placement
Fairness in exposure generally use rank-based exposure:

$$\mathcal{E}(q, d) = \mathbb{E}_y \left[ \sum_{k=1}^{K} \theta_k \mathbb{1}[y_k = d] \right] = \sum_{y \in \pi} \pi(y) \sum_{k=1}^{K} \theta_k \mathbb{1}[y_k = d].$$
Fairness in exposure generally use rank-based exposure:

\[
\mathcal{E}(q, d) = \mathbb{E}_y \left[ \sum_{k=1}^{K} \theta_k 1[y_k = d] \right] = \sum_{y \in \pi} \pi(y) \sum_{k=1}^{K} \theta_k 1[y_k = d].
\]

In this paper, we use a (novel) pairwise disparity-based fairness metric:

\[
\mathcal{F}(q) = \sum_{d_1 \in D} \sum_{d_2 \in D \setminus d_1} (\mathcal{E}(q, d_1) \rho_{d_2} - \mathcal{E}(q, d_2) \rho_{d_1})^2.
\]
Fairness in exposure generally use rank-based exposure:

$$\mathcal{E}(q, d) = \mathbb{E}_y \left[ \sum_{k=1}^{K} \theta_k 1[y_k = d] \right] = \sum_{y \in \pi} \pi(y) \sum_{k=1}^{K} \theta_k 1[y_k = d].$$

In this paper, we use a (novel) pairwise disparity-based fairness metric:

$$\mathcal{F}(q) = \sum_{d_1 \in D} \sum_{d_2 \in D \setminus d_1} (\mathcal{E}(q, d_1)\rho_{d_2} - \mathcal{E}(q, d_2)\rho_{d_1})^2.$$ 

PL-Rank can be applied to any rank-based exposure metric where:

$$\frac{\delta \mathcal{F}(q)}{\delta m} = \sum_{d \in D} \frac{\delta \mathcal{F}(q)}{\delta \mathcal{E}(q, d)} \frac{\delta \mathcal{E}(q, d)}{\delta m(d)}.$$

Method: PL-Rank for Fairness

Fairness in exposure generally use rank-based exposure:

\[ E(q, d) = \mathbb{E}_y \left[ \sum_{k=1}^{K} \theta_k \mathbb{1}[y_k = d] \right] = \sum_{y \in \pi} \pi(y) \sum_{k=1}^{K} \theta_k \mathbb{1}[y_k = d]. \]

PL-Rank can be used to optimize a rank-based exposure metric \( F \):

\[
\frac{\delta}{\delta m} F(q) = \sum_{d \in D} \left[ \frac{\delta}{\delta m} m(d) \right] \mathbb{E}_y \left[ \left( \sum_{k=\text{rank}(d,y)+1}^{K} \theta_k \left[ \frac{\delta F(q)}{\delta E(q,y_k)} \right] \right) \right]

+ \sum_{k=1}^{\text{rank}(d,y)} \pi(d \mid y_1:k-1) \left( \theta_k \left[ \frac{\delta F(q)}{\delta E(q,d)} \right] - \sum_{x=k}^{K} \theta_x \left[ \frac{\delta F(q)}{\delta E(q,y_x)} \right] \right). \]
Experimental Results
Results on Yahoo! Webscope

![Graph showing DCG@5 performance over minutes trained for different methods.](image)

- **DCG@5** performance over **Minutes Trained** for:
  - LambdaLoss
  - Policy Gradient
  - Placement Policy Gradient
  - PL-Rank-1
  - PL-Rank-2

The chart demonstrates the improvement in DCG@5 as the number of minutes trained increases for each of these methods.
Results on All Datasets

**MSLR-Web30k**

**Istella**

[Graph showing DCG@5 for different models over minutes trained]

<table>
<thead>
<tr>
<th>LambdaLoss</th>
<th>Policy Gradient</th>
<th>Placement Policy Gradient</th>
<th>PL-Rank-1</th>
<th>PL-Rank-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.4</td>
<td>7.6</td>
<td>7.8</td>
<td>8.0</td>
<td>8.2</td>
</tr>
<tr>
<td>17.5</td>
<td>18.0</td>
<td>18.5</td>
<td>19.0</td>
<td></td>
</tr>
</tbody>
</table>
Results: Exposure Fairness

Yahoo! Webscope

Minutes Trained

Mean Disparity

LambdaLoss, Policy Gradient, Placement Policy Gradient, PL-Rank-1, PL-Rank-2
Conclusion
Conclusion

PL-Rank: a novel LTR method for Plackett-Luce models:

- unbiased sample-based gradient estimation (no heuristic or bounding),
- computationally efficient (avoids combinatorial problems).
- applicable to relevance and fairness ranking metrics.

Continue our work: https://github.com/Harrie0/2021-SIGIR-plackett-luce
The **StochasticRank** algorithm (Ustimenko and Prokhorenkova, 2020) uses sampled noise to **stochastically smooth** a ranking function.

This algorithm has strong theoretical properties and could also be applied to Plackett-Luce models with comparable computational complexity.

Very promising direction for finding computationally efficient, effective and broadly applicable LTR.


