Policy Learning for Fairness in Ranking

Ashudeep Singh, Thorsten Joachims
(Cornell University)

NeurIPS 2019
For each query $q$

$$\tilde{x}_i = (x_{i,1}, x_{i,2}, \ldots x_{i,d})$$

True or Estimated relevances $\{\text{rel}_i\}$

Linear Program with Fairness constraints

Learning to Rank with Fairness constraints
Fair-PG-Rank: Learning-to-Rank with Fairness constraints

\[ \pi^*_\delta = \arg\max_{\pi} \mathbb{E}_{q \sim \mathcal{Q}} [U(\pi|q)] \quad \text{s.t.} \quad \mathbb{E}_{q \sim \mathcal{Q}} [\mathcal{D}(\pi|q)] \leq \delta \]

Empirical Risk Minimization:

\[ \hat{\pi}^*_\lambda = \arg\max_{\pi} \frac{1}{N} \sum_{q=1}^{N} U(\pi|q) - \lambda \frac{1}{N} \sum_{q=1}^{N} \mathcal{D}(\pi|q) \]

1. Ranking policy \( \pi \)
2. Utility of \( \pi \)
3. Disparity measure w.r.t. \( \pi \)
1. Ranking policy $\pi$

Sample using Plackett-Luce model

$\sim$ Ranking $r_i$
Sampling from $\pi$: Plackett-Luce Model

$p_i = \frac{e^{s_i}}{\sum_j e^{s_j}}$

$p'_i = \frac{e^{s_i}}{\sum_{j \neq i} e^{s_j}}, i \neq 2$

$P(\text{ranking} = \langle d_2, d_1, d_3, d_4 \rangle) = p_2 \times p'_1 \times p''_3 \times p''_4$
1. Ranking policy $\pi$

- Parameterized Model $h_{\theta}$
- Plackett-Luce (PL) Sampling

Sample using Plackett-Luce model

$\sim$

Ranking $r_i$
2. User Utility

\[ U(\pi|q) = \mathbb{E}_{r \sim \pi(r|q)}[\Delta(r, \text{rel}_q)] \]

For example:

- \( \Delta_{\text{DCG}}(r, \text{rel}_q) = \sum_{j=1}^{n_q} \frac{u(r(j)|q)}{\log(1+j)} \)
- \( \Delta_{\text{NDCG}}(r, \text{rel}_q) = \frac{\Delta_{\text{DCG}}(r, \text{rel}_q)}{\Delta_{\text{DCG}}(r^*, \text{rel}_q)} \)
3. Disparity Measures

- Exposure \( d_i = v_\pi(d_i) = \mathbb{E}_{r \sim \pi}[v_r(d_i)] \)
- Exposure \( G = v_\pi(G) = \sum_{d \in G} \text{Exposure}(d_i) \)
- Merit of a group \( M(G) = \sum_{d \in G} M(d_i) \)
- Group fairness disparity:
  \[
  D_{\text{group}}(\pi|q) = \max \left( 0, \frac{v_\pi(G_i)}{M_{G_i}} - \frac{v_\pi(G_j)}{M_{G_j}} \right), \text{ where } M_{G_i} \geq M_{G_j}
  \]
- Individual fairness disparity:
  \[
  D_{\text{individual}}(\pi|q) = \sum_{i,j} \max \left( 0, \frac{v_\pi(d_i)}{M_i} - \frac{v_\pi(d_j)}{M_j} \right) \quad \text{s.t. } M_i \geq M_j
  \]
Learning objective: \[ \hat{\pi}^*_\lambda = \arg\max_{\pi} \frac{1}{N} \sum_{q=1}^{N} U(\pi|q) - \lambda \frac{1}{N} \sum_{q=1}^{N} D(\pi|q) \]

Utility: \[ U(\pi|q) = \mathbb{E}_{r \sim \pi(r|q)} [\Delta(r, rel^q)] = \mathbb{E}_{r \sim \pi(r|q)} \left[ \sum_{j=1}^{n_q} \frac{u(r(j)|q)}{\log(1 + j)} \right] \]

Most previous methods optimize a smooth convex proxy of the utility. (SVMRank, ListNet, RankNet, Softrank etc.)

How do we perform a gradient descent update on utility as well as disparity?
PG-Rank: REINFORCE update for $\pi_\theta$

$$\nabla_\theta U(\pi_\theta | q) = \nabla_\theta \mathbb{E}_{r \sim \pi_\theta (r | q)} \Delta(r, \text{rel}^q)$$

Similarly for Disparity $\mathcal{D}_\pi$
Fair-PG-Rank: Summary

- **Input**: Queries w/ Candidate sets: features (+ relevances available during training)
- **Ranking policy**: A neural network scoring function + PL-sampling
- **Loss function**: Utility - $\lambda \cdot$ Disparity
- **Train**: Stochastic gradient descent
Experiment 1

- Each document $d_i$ represented by $(x_{i,1}, x_{i,2}) \in [0,3]^2$
- Two groups: $G_0, G_1$ (Ratio 4:1)
- $\text{rel}_i = x_{i,1} + x_{i,2}$ (True Model)
- $x_{i,2}$ corrupted for $G_1$ documents
- $h(x_i) = \theta_1 x_{i,1} + \theta_2 x_{i,2}$
- Train the model $(\theta_1, \theta_2)$ with different values of $\lambda$

A fair model only uses the $x_1$ feature
Experiment 2: Yahoo! Learning-to-Rank Challenge