Naver Labs Europe at TREC 2020 Fair Ranking Track

Till Kletti
NAVER LABS Europe
Meylan, France
till.kletti@naverlabs.com

Jean-Michel Renders
NAVER LABS Europe
Meylan, France
jean-michel.renders@naverlabs.com

ABSTRACT

In our participation to the TREC 2020 Fair Ranking Track, as Naver Labs Europe, we focused on the re-ranking task and we investigated the performance of a controller as a way to minimize unfairness over time, with minimal loss of utility.

We used a two-step approach, based on (1) a relevance probability estimator, and (2) a controller that aims to bring the actual exposure close to the target exposure.

This paper describes the components we designed in more detail. It contains a comparison of the performance of the controller to a baseline, which consists of a Plackett-Luce sampler. It also analyses the effect of the quality of the estimated relevance probabilities (closeness to the true binary relevance values) on the controller performance.

1 INTRODUCTION

This year, the TREC 2020 Fair Ranking Track introduces a new fairness metric, based on the differences between the system group expected exposures and the target group expected exposures, which has the particularity to mix fairness and utility into a single non-conflictual metric. Indeed, for a given query, if the relevance values of documents are known, an optimal policy exists that simultaneously offers maximum utility in the PRP (*Probability Ranking Principle*) sense and optimal fairness according to this year's fairness metric, at least after a periodic number of repetitions. In case of binary relevance values, this policy consists in cycling over all $n_r!(N-n_r)!$ rankings for the given query, where n_r is the number of relevant documents and N is the total number of documents (see [1]).

Overall, in this year's TREC Fair Ranking Track, the main idea we wanted to test and evaluate in a realistic setting was the design of a controller that optimizes at any time the system fairness (as defined by the metric of this year), while dealing with the fact that the relevance values are actually unknown and can only be approximated based on an effective retrieval algorithm. We focused on the re-ranking task, leaving the full retrieval task for future work.

In a nutshell, our solution is made of two independent modules:

- §2 **Indexing and Re-ranking**: This module estimates for each query the relevance probabilities, i.e. the probabilities of a document being relevant for the query.
- §3 **Fairness Controller**: This module uses the relevance probabilities to bring the expected actual exposure close to the expected target exposure.

Note that, since the unfairness is defined by the TREC guidelines as an average over the query-level metric, we do not have to worry about amortization over different queries and can treat each query independently.

In the following, we describe our document indexing and re-ranking process. Then, we present our fairness controller, followed by an analysis of our obtained results.

2 INDEXING & RE-RANKING

The main goal of this component is to estimate accurately the relevance probabilities of every document with respect to the query. As far as implementation choices are concerned, we decided to rely on rather simple indexing and retrieval techniques. This decision was taken assuming that the training set information (queries, document subsets and relevance signals) was representative of the evaluation set. The motivations for these choices are the following: (1) we are dealing with a re-ranking task, which is much less complicated than providing a full ranking across the whole collection; (2) queries and documents are multi-lingual and, even if we could have relied on a language detector and have separate pre-processing chains per language, we preferred to adopt a language-agnostic approach; (3) we observed that the relevance signals seem to come from click data, for which we had no knowledge about the context (rank of the document in the initial SERP, time of the query, etc.), so that we can consider these signals as biased and noisy, with little hope that a complex retrieval algorithm will be robust enough to give results significantly better than a standard method such as BM25[1] .

In practice, as we didn't rely on any query expansion and as we were targeting only the re-ranking task, we only indexed the subset of the collection that contains the documents to be re-ranked. To be as language-agnostic as possible, we used a simple tokenizer based on usual white space characters and punctuation, without any stop-word removal.

As retrieval model, we used a combination of BM25 [5] and word embedding-based approaches [3] for computing basic relevance scores. In a first retrieval model (called `ModelB` hereafter), we simply used a linear combination of the two scores to compute a fused score; the corresponding weight is determined by a line search in a cross-validation setting. In a second retrieval model (called `ModelA` hereafter), these intermediate relevance scores were combined with metadata (recency and number of citations) through a Gradient-Boosted-Tree classifier, trained with a pointwise log-likelihood objective function. The hyper-parameters used to train this classifier were identified by cross-validation on the training queries. The output of these two retrieval models provides us with estimated relevance scores that we have to transform into relevance probabilities. This calibration step is realised by Isotonic Regression, as described in [4]. After this calibration step, the retrieval component output consists then of a vector of estimated relevance probabilities,

[1] Actually, we tried some BERT-based retrieval models, pre-trained on the MS-MARCO dataset and fine-tuned with the (rather small) training set of this track, but the retrieval performance was disappointing with respect to a simple BM25 model.

denoted as $\rho = (\rho_i)_{i\in\{1,N\}}$, for each of the N documents associated to the considered query.

3 FAIR CONTROLLER

We consider the true unknown relevance values to be realizations of Bernoulli random variables with parameters ρ, computed as in section 2. Given a query, with N documents and relevance probabilities $\rho = (\rho_i)_{i\in\{1,N\}}$, we assume the actual relevance $r_i \sim \text{Ber}(\rho_i)$ independently for each document i. As the document relevance values are unknown, their target exposures $\mathcal{E}_i^*$ as defined in [1] are unknown as well. But we can compute their expected values, given the relevance probabilities. More precisely, the expected target exposure of a document i can be expressed as

$$\mathbb{E}_{r\sim\text{Ber}(\rho)}\left[\mathcal{E}_i^*\right] = \sum_{s=0}^{N-1} \text{PB}(s|\rho_{-i})\left(\rho_i\mu_{s+1} + (1-\rho_i)\nu_s\right), \quad (1)$$

where ρ_{-i} denotes the vector of parameters ρ excluding its i^{th} element, $\text{PB}(\cdot|\rho)$ denotes the density of a Poisson-Binomial distribution with parameters ρ, and where μ_s, ν_s respectively denote the target exposure of a relevant document and the target exposure of an irrelevant document, given that s amongst N documents are relevant (see [1] for details of their computation). The exact computation of the probability mass function of a Poisson-Binomial is no trivial matter. We used a pre-coded method described in [2] and implemented in [6]. The quantities $\mathbb{E}[\mathcal{E}_i^*]$ need to be computed once for each query. Document expected target exposures are then propagated to expected target exposures per producer by simple summation, as defined in the track guidelines. Then they need only to be multiplied by t to get their value after t repetitions of the query. We will denote the producer expected target exposure after t repetitions as $\hat{\mathcal{E}}_{p,t}^*$, and we have:

$$\hat{\mathcal{E}}_{p,t}^* = t \sum_{i \text{ produced by } p} \mathbb{E}[\mathcal{E}_i^*] \quad (2)$$

Due to the lack of knowledge of the true relevance values, the actual exposures are unknown as well. The computation of the expected actual exposure $\mathbb{E}[\mathcal{E}_i]$ is more straightforward than the one of the expected target exposure. Indeed, assuming the linearity of f – the function that maps the relevance value into a probability of the user being satisfied, as defined in the track guidelines –, one can show that given a ranking π mapping a document to its rank, we have

$$\mathbb{E}_{r\sim\text{Ber}(\rho)}[\mathcal{E}_i|\pi] = \gamma^{\pi(i)-1} \prod_{j=1}^{\pi(i)-1} \left(1 - f(\rho_{\pi^{-1}(j)})\right). \quad (3)$$

Expected document exposures are then propagated to expected producer exposures by simple summation, as defined in the track guidelines. Since the group memberships are unknown, we decide to control for the exposure at producer level. Indeed a low unfairness at producer level implies a low unfairness at group level. After t repetitions of the query, as each repetition uses in general a different ranking $\pi^{t'}$ ($t' \in \{1,\dots,t\}$), the actual expected exposure of producer p at repetition/time t is simply:

$$\hat{\mathcal{E}}_{p,t} = \sum_{t'=1}^{t} \mathbb{E}_{r\sim\text{Ber}(\rho)}[\mathcal{E}_p|\pi^{t'}] \quad (4)$$

Algorithm 1 shows the layout of our simple controller. Given a query and a history of delivered rankings, we define the *advantage* $A_t(p)$ of a producer p at repetition t as

$$A_t(p) = (\hat{\mathcal{E}}_{p,t-1} - \hat{\mathcal{E}}_{p,t-1}^*)^2 \,\text{sign}(\hat{\mathcal{E}}_{p,t-1} - \hat{\mathcal{E}}_{p,t-1}^*), \quad (5)$$

The advantage $A_t(i)$ of a document i is defined to be the arithmetic mean of the advantages of its producers. The advantage is a signed real number expressing how much a producer has been advantaged (if $A_t(p) > 0$) or disadvantaged (if $A_t(p) < 0$) in the past (i.e. up to repetition $t-1$). The advantage is the equivalent of the error (between the system output and the target) in standard control theory. When the history is empty, both target and actual exposure coincide (both are 0), so all advantages are zero. Given a query and a history of delivered rankings, we define the output of our controller at repetition/time t by the *scores* $(h_{i,t})_{i\in\{1,N\}}$ for each document as

$$h_{i,t} = \theta\rho_i + (1-\theta)A_t(i), \quad (6)$$

with $\theta \in (0,1)$. In standard control theory, this corresponds to a P-controller (or proportional controller), as only the term proportional to the error is included in the control signal. Note that we have one controller per query, as the task defines fairness metric at the individual query level. Our ranking policy simply consists in ordering the documents by decreasing score, as output by the controller at each repetition t. Ties are broken randomly.

Algorithm 1 Outline of the controller for a unique query q. P is the set of producers, D is the set of documents, $\hat{\mathcal{E}}_{p,t}$ (resp. $\hat{\mathcal{E}}_{p,t}^*$) denotes the total expected actual (resp. target) exposure at repetition t

procedure `Fair_Controller`
  $A_1(p) \leftarrow 0,\ \forall p \in P$ ▷ Initialise producer advantages to 0
  $\forall p \in P,\ \hat{\mathcal{E}}_{p,0} \leftarrow 0,\ \hat{\mathcal{E}}_{p,0}^* \leftarrow 0$ ▷ Initialise exposures to 0
  for $t = 1$ to $t = n$ **do**
    $\forall d \in D,\ A_t(d) \leftarrow \texttt{mean}\{A_t(p)|p \text{ produces } d\}$ ▷ Compute the document advantages
    $(h_{i,t})_{i\in\{1,N\}} \leftarrow \begin{pmatrix}\rho_1 & -A_t(1)\\ \vdots & \vdots\\ \rho_N & -A_t(N)\end{pmatrix}\begin{pmatrix}\theta\\ 1-\theta\end{pmatrix}$ ▷ Compute the adjusted scores
    $\pi \leftarrow (h_{1,t},\dots,h_{N,t})$ ▷ Sort the scores
    $\forall p \in P,\ \hat{\mathcal{E}}_{p,t} \leftarrow \hat{\mathcal{E}}_{p,t-1} + \sum_{i \text{ produced by} p}\mathbb{E}[\mathcal{E}_{i,t}|\pi]$ ▷ Update the expected actual exposures using equation (3)
    $\forall p \in P,\ \hat{\mathcal{E}}_{p,t}^* \leftarrow \hat{\mathcal{E}}_{p,t-1}^* + \sum_{i \text{ produced by} p}\mathbb{E}\left[\mathcal{E}_{i,t}^*\right]$ ▷ Update the expected target exposure using equation (1)
    $\forall p \in P,\ A_{t+1}(p) \leftarrow (\hat{\mathcal{E}}_{p,t} - \hat{\mathcal{E}}_{p,t}^*)^2 \,\text{sign}(\hat{\mathcal{E}}_{p,t} - \hat{\mathcal{E}}_{p,t}^*)$ ▷ Update the producer advantages
  end for
end procedure

4 SUBMITTED RUNS

We submitted five runs to the TREC Fair Ranking Track (re-ranking task):

- `ModelA_99_1`: Retrieval Model A (with recency and citation metadata) combined with the fair controller with $\theta = 0.99$

- `ModelA_9_1`: Retrieval Model A using the fair controller with $\theta = 0.9$
- `PL` $\tau = 0.05$: This is our baseline method: it consists in sampling rankings from a Plackett-Luce distribution with parameters $(\rho_i)_{i\in\{1,N\}}$ and with temperature $\tau = 0.05$
- `ModelB_99_1`: Retrieval Model B (without recency and citation metadata) combined with the fair controller with $\theta = 0.99$
- `ModelB_9_1`: Retrieval Model B using the fair controller with $\theta = 0.9$

5 RESULTS

In this section, we provide a concise analysis of the performance of our 5 submitted runs as reported by the official track metrics. Each query is repeated $T = 150$ times and we report performances averaged over 200 queries.

Table 1 presents the average performance of each of our runs on the training set in terms of the norm of the controlled differences $\|\mathbb{E}[\mathcal{E}^*_{\mathcal{P},T}] - \mathbb{E}[\mathcal{E}_{\mathcal{P},T}]\|$ and in terms of the actual producer-level unfairness $\|\mathcal{E}^*_{\mathcal{P},T} - \mathcal{E}_{\mathcal{P},T}\|$. The index $\mathcal{P}$ indicates that the exposures are considered vectors with one element per producer at time T. A run with $\theta = 1$ is also provided for comparison; it is equivalent to a *PRP* policy using the estimated relevance probabilities ρ.

On the training set the controller performs better than both the simple *PRP* policy based on the ρ_i and the Plackett-Luce policy, although the difference w.r.t Plackett-Luce is not very big. Amongst the controller methods, model A slightly outperforms Model B, and a slight advantage can be obtained by choosing $\theta = 0.99$ instead of $\theta = 0.9$.

Table 2 presents the performance of our submitted runs on the evaluation set in terms of the norm of the controlled difference $\|\mathbb{E}[\mathcal{E}^*_{\mathcal{P},T}] - \mathbb{E}[\mathcal{E}_{\mathcal{P},T}]\|$ and in terms of the actual group-level unfairness. On the evaluation set, no significant performance difference could be detected between either of our submitted run, although the TREC mean is clearly outperformed[2].

Figure 1 has been obtained by applying the controller and the Plackett-Luce policy to the true relevance values r_i blurred towards the estimated relevance probabilities ρ_i by a "blur factor" in $[0, 1]$. This was done only for the training queries, for which we have the true relevance information. A blur factor of $\lambda \in [0, 1]$ means that the simulated relevance probabilities for each document i were taken to be equal to $(1 - \lambda)r_i + \lambda\rho_i$. The results show that our controller does not provide any advantage with respect to a Plackett-Luce policy except when the relevance probabilities are very certain, i.e. when they are close enough to their true binary relevance values, in which case the controller tends to achieving almost zero unfairness.

6 CONCLUSIONS

We separated the problem into two subproblems: (§2) estimation of the relevance probabilities, and (§3) design of a controller bringing the expected actual exposures close to the expected target exposures at the producer-level.

We compared the controller to a baseline Plackett-Luce (PL) policy. We found that the controller slightly outperformed PL on the training data in terms of producer-level unfairness. But we did

[2]Statistical significance tests based on a paired t-test confirmed this observation with a p-value less than 10^{-12}.

Table 1: Average training results in terms of unfairness (lower is better). Best values are highlighted in bold. $T = 150$

run $(\theta, 1-\theta)$	$\|\mathbb{E}[\mathcal{E}^*_{\mathcal{P},T}] - \mathbb{E}[\mathcal{E}_{\mathcal{P},T}]\|$	$\|\mathcal{E}^*_{\mathcal{P},T} - \mathcal{E}_{\mathcal{P},T}\|$
Model A $(1, 0)$	230.8762	274.1540
Model A $(0.99, 0.01)$	28.3936	**156.1653**
Model A $(0.9, 0.1)$	**27.5066**	156.2799
Model A PL $\tau = 0.05$	75.8210	168.1693
Model B $(1, 0)$	236.5348	281.6143
Model B $(0.99, 0.01)$	28.4934	157.0478
Model B $(0.9, 1)$	27.7479	158.8281

Table 2: Average evaluation results in terms of unfairness (lower is better). Best values are highlighted in bold. An index $\mathcal{P}$ (resp. $\mathcal{G}$) means the exposures are computed in the producer (resp. group) space. $T = 150$

run $(\theta, 1-\theta)$	$\|\mathbb{E}[\mathcal{E}^*_{\mathcal{P},T}] - \mathbb{E}[\mathcal{E}_{\mathcal{P},T}]\|$	$\|\mathcal{E}^*_{\mathcal{G},T} - \mathcal{E}_{\mathcal{G},T}\|$
Model A $(0.99, 0.01)$	27.9860	87.5628
Model A $(0.9, 0.1)$	**26.9547**	**87.4756**
Model A PL $\tau = 0.05$	78.0179	87.5225
Model B $(0.99, 0.01)$	28.1344	88.4720
Model B $(0.9, 1)$	27.1246	88.5984
TREC mean	-	113.5034

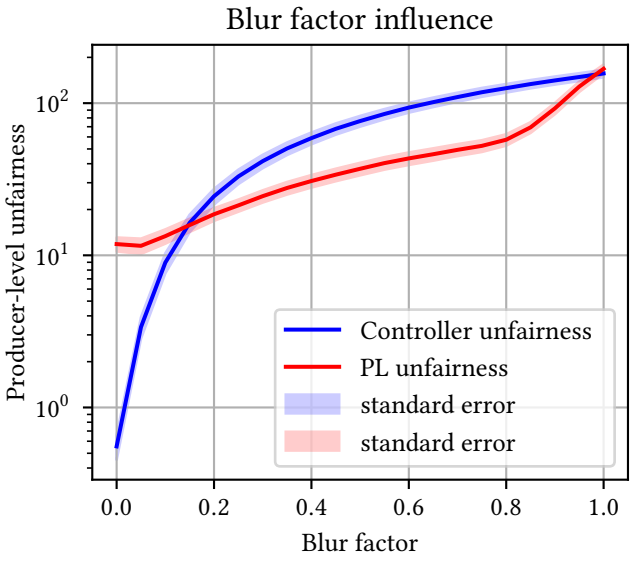

Figure 1: Analysis of the effect of "blurring" the true relevance values towards the relevance probabilities as $(1-\lambda)r_i + \lambda\rho_i$ with blur factor λ ranging from 0 to 1. Note that the y-scale is logarithmic.

not find sufficient evidence on the evaluation set to conclude that the controller outperforms PL in terms of group level fairness.

However, we found that the controller significantly outperforms PL, when the relevance probabilities tend to be very close to the true binary relevance values, with the controller tending to a near perfect fairness performance.

REFERENCES

[1] Fernando Diaz, Bhaskar Mitra, Michael D. Ekstrand, Asia J. Biega, and Ben Carterette. 2020. Evaluating Stochastic Rankings with Expected Exposure. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Oct. 2020), 275–284. https://doi.org/10.1145/3340531.3411962 arXiv: 2004.13157.

[2] Yili Hong. 2013. On computing the distribution function for the Poisson binomial distribution. *Computational Statistics & Data Analysis* 59 (March 2013), 41–51. https://doi.org/10.1016/j.csda.2012.10.006

[3] Bhaskar Mitra, Eric Nalisnick, Nick Craswell, and Rich Caruana. 2016. A Dual Embedding Space Model for Document Ranking. *arXiv:1602.01137 [cs]* (Feb. 2016). http://arxiv.org/abs/1602.01137 arXiv: 1602.01137.

[4] Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting Good Probabilities with Supervised Learning. In *Proceedings of the 22nd International Conference on Machine Learning* (Bonn, Germany) *(ICML '05)*. Association for Computing Machinery, New York, NY, USA, 625–632. https://doi.org/10.1145/1102351.1102430

[5] K. Sparck Jones, S. Walker, and S. E. Robertson. 2000. A probabilistic model of information retrieval: development and comparative experiments: Part 1. *Information Processing & Management* 36, 6 (Nov. 2000), 779–808. https://doi.org/10.1016/S0306-4573(00)00015-7

[6] tsakim. 2020. tsakim/poibin. https://github.com/tsakim/poibin original-date: 2016-05-02T08:40:41Z.