Sampling Bias Due to Near-Duplicates in Learning to Rank

Maik Fröbe[1] Janek Bevendorff[2] Jan Heinrich Reimer[1] Martin Potthast[3] Matthias Hagen[1]

[1]Martin-Luther-Universität Halle-Wittenberg, [2]Bauhaus-Universität Weimar, [3]Leipzig University

ABSTRACT

Learning to rank (LTR) is the de facto standard for web search, improving upon classical retrieval models by exploiting (in)direct relevance feedback from user judgments, interaction logs, etc. We investigate for the first time the effect of a sampling bias on LTR models due to the potential presence of near-duplicate web pages in the training data, and how (in)consistent relevance feedback of duplicates influences an LTR model's decisions. To examine this bias, we construct a series of specialized LTR datasets based on the ClueWeb09 corpus with varying amounts of near-duplicates. We devise worst-case and average-case train/test splits that are evaluated on popular pointwise, pairwise, and listwise LTR models. Our experiments demonstrate that duplication causes overfitting and thus less effective models, making a strong case for the benefits of systematic deduplication before training and model evaluation.

ACM Reference Format:
Maik Fröbe, Janek Bevendorff, Jan Heinrich Reimer, Martin Potthast, and Matthias Hagen. 2020. Sampling Bias Due to Near-Duplicates in Learning to Rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30,2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages.
https://doi.org/10.1145/3397271.3401212

1 INTRODUCTION

The loss functions used in learning to rank, at present, do not take into account inherent similarities between ranked documents. Still, it is well-known that web crawls contain a large number of pages highly similar to others [9], and that such near-duplicate documents obtain identical relevance labels with a high probability [2]. Training and testing LTR models on non-deduplicated web data therefore causes implicit oversampling. The adverse effects of such uncontrolled oversampling before the data is partitioned into train/test splits are known and have only recently been demonstrated again [23]. Furthermore, the overrepresentation of some training examples over others increases the chances of overfitting and the comparably complex web search pipeline opens up many additional opportunities for inducing and reinforcing biases. A duplicated document is first created, then crawled, and finally parsed into features for ranking. Each of these independent steps can be biased on its own [24, 26]. Hence, the ramifications of the presence of near-duplicates on LTR models are unclear.

We create a dedicated LTR dataset from the ClueWeb09 corpus, using the relevance labels of the TREC 2009–2012 Web Tracks (Section 3), and carry out a first empirical analysis of the biases and performance deficits introduced by near-duplicates (Section 4). Our focus in this short paper lies primarily on redundantly crawled Wikipedia articles like the ones shown in Figure 1. Wikipedia uses canonical link relations[1] to canonicalize synonymous concepts, redirecting users, and generating a corresponding hint into the article text. However, neither these nor any other forms of deduplication were considered during the ClueWeb09 crawling process, and the pooling of relevance assessments in the Web Tracks causes both documents in Figure 1 to be judged as relevant to the query `designer dog breeds`. This is not necessarily a problem for TREC evaluations, since different runs might have reported the two versions of the document. Still, near-duplicates in a single result list are not really helping the user, and groups of near-duplicates affect the training of models. When virtually the same document occurs multiple times during the algorithm-specific loss minimization and neither the training labels nor the loss function itself penalize redundant training input [2], convergence becomes either more difficult (given different judgments for basically the same content), or potentially too easy (given identical judgments for near-duplicates). As a result, duplicates of already seen results will appear further down in the list and commonly applied effectiveness measures, such as nDCG, might even "reward" more than one copy of (very) relevant near-duplicates in a ranking.

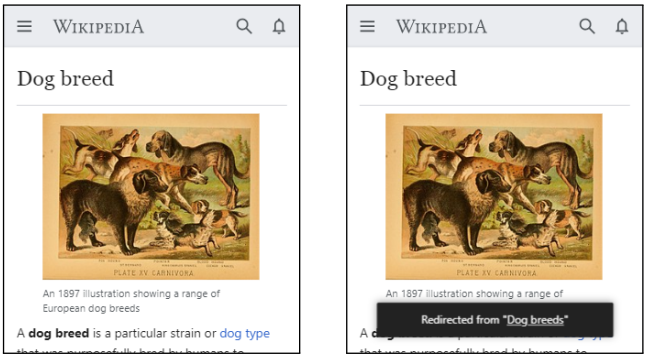

Figure 1: Example of near-duplicate Wikipedia articles, one returned for *Dog breed* (left), one for *Dog breeds* (right). Both are identical except for the redirect message.

Our comprehensive comparison of various LTR algorithms trained on the Web Track topics with the highest levels of redundancy demonstrates a hidden model bias (Section 4.2). We contrast a worst-case scenario with an excessive number of relevant duplicates during training with results from randomly drawn average-case train/test splits, which eventually leads us to an immediate practical insight: Deduplicating training and test data helps to develop more robust LTR models. The software we used is based on open-source tools [8, 28] and is available online together with all data.[2]

[1]https://en.wikipedia.org/wiki/Canonical_link_tag
[2]https://github.com/webis-de/SIGIR-20

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401212

Table 1: (a) Number of near-duplicates from the most redundant domains (Red.), and proportion of relevant documents within duplicates (Rel.). (b) Judgement consistency for near-duplicate documents. (c) Labeled documents in train/test splits, number of redundant documents, and proportion of relevant documents within duplicates. (d) Learning to rank features computed for judged documents from ClueWeb09. Text features are calculated on title, main content, body, and anchor texts separately [12].

(a) *Redundant domains*

Domain	Tag*	Red.	Rel.
wikipedia.org	Research	7,225	32 %
memoryx.net	Tech	166	0 %
supercrawler.com	Tech	98	29 %
meetup.com	Social	82	0 %

(b) *Judgement consistency*

Track	Incons. Docs.
Web 2009	2.59 %
Web 2010	1.88 %
Web 2011	1.35 %
Web 2012	1.39 %

(c) *Train/test splits*

Split	Training Labels			Test Labels		
	Count	Red.	Rel.	Count	Red.	Rel.
Worst-case scenario	16,675	2,474	50 %	9,994	1,298	4 %
5-fold cross validation 1	21,337	3,004	33 %	5,309	729	32 %
5-fold cross validation 2	21,062	3,000	32 %	5,584	733	39 %
5-fold cross validation 3	21,960	3,013	34 %	4,686	720	27 %
5-fold cross validation 4	21,642	2,969	33 %	5,004	764	35 %
5-fold cross validation 5	20,583	2,946	33 %	6,063	787	33 %

(d) *Learning to rank features*

Text		Web Graph†	
Description	Count	Description	Count
Term frequency	4	URL length	1
TF · IDF	4	No. of slashes in URL	1
BM25 score	4	PageRank‡	1
F2 exp score	4	SpamRank [5]	1
F2 log score	4	No. of inlinks	1
QL score	4	No. of outlinks	1
QLJM score	4		
PL2 score	4		
SPL score	4		
Σ Total			42

*Domain tags are retrieved from OpenDNS: https://domain.opendns.com
†By CMU: https://lemurproject.org/clueweb09/webGraph.php
‡https://lemurproject.org/clueweb09/pageRank.php

2 RELATED WORK

Bernstein and Zobel [2] discovered that 16 % of all relevant documents in runs submitted to the TREC Terabyte Track 2004 [4] are near-duplicates. To cleanse the retrieval evaluation from this redundancy, they introduced the novelty principle, which states that a document—though relevant in isolation—becomes irrelevant if another equivalent document has been ranked higher. Their study shows that the application of the novelty principle decreases MAP scores by 20 % on average. The novelty principle's effect on LTR models remains unclear, since none were submitted to this track.

It took several more years for LTR to gain traction, and tailored datasets have simplified the development of LTR algorithms since [3, 19, 21, 22]. From these, the detection of near-duplicates is possible only on the LETOR [3] and the MS MARCO [19] datasets, which supply documents. Unfortunately, both datasets are judged sparsely, reducing their usefulness for a deeper analysis in this regard.

Nevertheless, Minka and Robertson identified a selection bias in the LETOR datasets that could lead LTR models to perform worse than baseline models [18]. Further kinds of selection [3, 20, 25, 29], label balance [13], and implicit user feedback biases [1, 14, 26] have been exposed since, but the selection bias caused by the presence of near-duplicates under Bernstein and Zobel's novelty principle has been neglected so far [11]. Moreover, despite all previous research, many biases remain unmitigated in the datasets, posing a risk to the fairness of retrieval systems [6]. In this paper, we take another step towards understanding and managing these risks.

3 DUPLICATE-AWARE LTR DATASETS

For our experiments, we construct a series of tailored LTR datasets to study the effect of the presence or absence of near-duplicate documents. We derive these datasets from the ClueWeb09 corpus, a general-purpose web crawl that has been frequently used in LTR research [15, 16, 30]. Despite its age, we prefer the ClueWeb09 corpus over more recent crawls like ClueWeb12, since it comes with a significantly higher number of topics and corresponding relevance judgments that can be used for training and testing.

For an initial assessment of the overall amount of duplication among the judged ClueWeb09 documents, we employ the lossless fingerprint similarity S_3 of Bernstein and Zobel [2] using word-8-grams. An S_3 score of 0 indicates no overlap between documents and an S_3 score of 1 means equality. We sample 100 document pairs that uniformly cover S_3 scores between 0.4 and 1 for a manual review and find that pairs with scores above 0.84 are near-duplicates with a precision of 0.95. Table 1a shows the domains with the highest number of near-duplicates per domain. The most redundant domain, wikipedia.org, has 7,225 redundant documents, of which a remarkable portion of 32 % is judged as relevant. This is six percentage points larger than the relevant portion of all Wikipedia documents and ten points larger than the relevant portion of all labeled ClueWeb09 documents. Other domains provide significantly fewer duplicates—a fact we could not sufficiently explain so far. Table 1b confirms that near-duplicates receive the same relevance label with almost certain probability (above 97 %), which corroborates the study of Bernstein and Zobel [2].

Since the largest amount of duplicates stems from wikipedia.org, we focus our study on this domain, using Wikipedia's canonical link relations as the ground truth to obtain large numbers of near-duplicates with near-perfect precision. For each group of duplicates, we designate one representative document based on the article's canonical link relation, which is unambiguous in 60 % of the cases (like in Figure 1). For the remaining documents, we choose the most frequent canonical URL as the most likely candidate. Based on a manual review of a sample of 100 ambiguous cases, we conclude that ambiguities arise largely from missing canonical URLs (which Wikipedia introduced only halfway through the ClueWeb09 crawl).

Table 2: *nDCG@20* performance on trained models and mean first rank of irrelevant documents from `wikipedia.org`. Training sets are either (1) unmodified (100 % redundancy), (2) deduplicated (0 % redundancy), or apply the novelty principle (*NOV*).

	Algorithm	nDCG@20 Performance									Mean First Rank		
		Duplicates Unmodified			Duplicates Irrelevant			Duplicates Removed			Irrel. Wiki. Docs.		
		100 %	0 %	*NOV*	100 %	0 %	*NOV*	100 %	0 %	*NOV*	100 %	0 %	*NOV*
Worst-Case Scenario	BM25	0.066	—	—	0.059	—	—	0.079	—	—	83	—	—
	Coor. Ascent	0.164	$0.150^{\downarrow 0.1}$	$0.177^{\uparrow 0.1}$	0.155	$0.142^{\downarrow 0.1}$	$0.170^{\uparrow 0.1}$	0.186	$0.167^{\downarrow 0.1}$	$0.190^{\uparrow 0.0}$	19	$\mathbf{43}^{\uparrow \mathbf{0.6}}$	$\mathbf{48}^{\uparrow \mathbf{0.8}}$
	LambdaMart	0.114	$0.138^{\uparrow 0.2}$	$\mathbf{0.168}^{\uparrow \mathbf{0.4}}$	0.120	$0.141^{\uparrow 0.2}$	$\mathbf{0.167}^{\uparrow \mathbf{0.3}}$	0.135	$0.156^{\uparrow 0.2}$	$\mathbf{0.186}^{\uparrow \mathbf{0.4}}$	19	$17^{\downarrow 0.1}$	$22^{\uparrow 0.1}$
	RankBoost	0.165	$0.174^{\uparrow 0.1}$	$0.176^{\uparrow 0.1}$	0.151	$0.166^{\uparrow 0.1}$	$0.155^{\uparrow 0.0}$	0.189	$0.193^{\uparrow 0.0}$	$0.179^{\downarrow 0.1}$	28	$39^{\uparrow 0.3}$	$34^{\uparrow 0.1}$
	Regression	0.121	$0.121^{\uparrow 0.0}$	$\mathbf{0.179}^{\uparrow \mathbf{0.3}}$	0.108	$0.125^{\uparrow 0.1}$	$\mathbf{0.173}^{\uparrow \mathbf{0.4}}$	0.139	$0.143^{\uparrow 0.0}$	$\mathbf{0.187}^{\uparrow \mathbf{0.3}}$	14	$10^{\downarrow 0.2}$	$11^{\downarrow 0.2}$
5-Fold Cross Validation	BM25	0.146	—	—	0.113	—	—	0.150	—	—	83	—	—
	Coor. Ascent	0.264	$0.268^{\uparrow 0.0}$	$\mathbf{0.229}^{\downarrow \mathbf{0.2}}$	0.177	$\mathbf{0.225}^{\uparrow \mathbf{0.3}}$	$\mathbf{0.219}^{\uparrow \mathbf{0.3}}$	0.247	$\mathbf{0.273}^{\uparrow \mathbf{0.1}}$	$0.249^{\uparrow 0.0}$	29	$\mathbf{40}^{\uparrow \mathbf{0.3}}$	$\mathbf{47}^{\uparrow \mathbf{0.5}}$
	LambdaMart	0.252	$\mathbf{0.236}^{\downarrow \mathbf{0.1}}$	$\mathbf{0.215}^{\downarrow \mathbf{0.2}}$	0.196	$0.201^{\uparrow 0.0}$	$\mathbf{0.214}^{\uparrow \mathbf{0.1}}$	0.243	$0.238^{\downarrow 0.0}$	$\mathbf{0.230}^{\downarrow \mathbf{0.1}}$	31	$\mathbf{24}^{\downarrow \mathbf{0.2}}$	$33^{\uparrow 0.1}$
	RankBoost	0.278	$0.263^{\downarrow 0.1}$	$0.262^{\downarrow 0.1}$	0.199	$\mathbf{0.228}^{\uparrow \mathbf{0.2}}$	$0.204^{\uparrow 0.0}$	0.272	$0.279^{\uparrow 0.0}$	$0.269^{\downarrow 0.0}$	41	$\mathbf{48}^{\uparrow \mathbf{0.2}}$	$46^{\uparrow 0.1}$
	Regression	0.230	$\mathbf{0.190}^{\downarrow \mathbf{0.2}}$	$0.223^{\downarrow 0.0}$	0.142	$\mathbf{0.169}^{\uparrow \mathbf{0.2}}$	$\mathbf{0.191}^{\uparrow \mathbf{0.4}}$	0.211	$\mathbf{0.196}^{\downarrow \mathbf{0.1}}$	$\mathbf{0.224}^{\uparrow \mathbf{0.1}}$	26	$\mathbf{32}^{\uparrow \mathbf{0.2}}$	$\mathbf{39}^{\uparrow \mathbf{0.3}}$

The 60 Web Track topics with the overall highest level of redundancy are chosen for building six train/test splits (Table 1c). Serving as an empirical upper bound for the impact of duplication on LTR, one of these is sampled as a worst-case split, in which the 40 most redundant topics with the highest ratio of duplicates judged as relevant, form the training set. The remaining 20 topics with a duplicate relevance ratio of only 4 % form the test set. The other five splits are created using 5-fold cross validation with balanced amounts of duplication and even relevance ratios across training and test.

Table 1d shows the 42 LTR features (similar to LETOR [21]) that we calculate for the judged documents from the 60 selected topics. We remove documents with BM25@body of zero from the training to prevent the selection bias of earlier LETOR versions [18].

4 EXPERIMENTAL ANALYSIS

All LTR algorithms implemented in RankLib [8] are compared on our LTR datasets to investigate the effect of duplicates on LTR models regarding their performance, and possible biases towards these duplicates. Some LTR algorithms are not deterministic, so that the experiments were repeated 5 times, reporting the averages. We discuss the results of Coordinate Ascent [17], LambdaMart [27], RankBoost [10], and linear regression in detail, all of which receive the best nDCG@20 scores on our LTR datasets. Their parameters were left at their defaults, and no countermeasures to overfitting (like regularization) were applied.

The LTR models are trained with three different deduplication strategies: (1) no deduplication (100 % baseline), (2) full deduplication by retaining only the representative document of each group of duplicates (0 % duplicates), and (3) penalization of relevance labels for duplicates similar to the novelty principle (*NOV*). The 0 % strategy comes at the general disadvantage of discarding 14 % of the training data (see Table 1c). *NOV* adds an additional boolean feature indicating if a document is canonical and the training labels are modified to decrease the relevance of non-canonical duplicates by 90 % to create a target ranking that lists relevant documents above relevant duplicates followed by irrelevant documents. The *NOV* strategy remedies the shortcoming of the 0 % strategy and leverages all training data, but breaks the consistency of the labels.

4.1 Performance Impact of Redundancy

For each of the three training data deduplication strategies, we report the impact of duplicates on model performance using three different evaluation methodologies: (1) duplicates retain their relevance in the test set (as a baseline), (2) duplicates are irrelevant (Bernstein and Zobel's novelty principle [2]), and (3) duplicates are irrelevant and removed from the test set. We see the latter as the most realistic scenario, which not only models the novelty principle, but also applies deduplication as a preprocessing step. All evaluations are conducted with `gdeval`.[3] In each iteration, we manipulate the qrels for the runs according to the strategy at hand. For evaluation under the novelty principle, we manipulate the relevance judgments so that the highest-ranked document from each group of duplicates receives the relevance of the representative document of that group, while all other documents become irrelevant.

No existing run from the Web Tracks is suited for comparison, since none of them scores all documents in our dataset. Therefore, we contrast the LTR algorithms of RankLib with sorting the documents by descending BM25@body to provide a reasonable overall baseline. All LTR algorithms improve upon our BM25 baseline. We contrast the two deduplication strategies with full redundancy and report effect sizes (Cohen's d) and statistical significance ($p \leq 0.05$).

The upper half of the "nDCG@20 Performance" columns in Table 2 shows the performance for RankLib's four best-performing algorithms on our worst-case train/test split. LambdaMart fits the training data very closely and of all algorithms generalizes the least to the worst-case test set. All algorithms perform better under one of the two training data deduplication strategies and overall better when duplicates in the test set were either marked as irrelevant or removed altogether. This is expected considering the duplicate relevance disparity of 50 % in the training and only 4 % in the test set. In the lower half of the table, averages of the results on the five more realistic cross validation splits are shown. For these, we

[3] By the Web Track organizers: https://trec.nist.gov/data/web/10/gdeval.pl

find the opposite effect that deduplication tends to worsen model performance when duplicates retain their original relevance in the test set. If duplicates become irrelevant by application of the novelty principle, models trained without deduplication see decreased nDCG@20 scores by 30 % on average. Models trained with deduplication, on the other hand, perform (often significantly) better, which is in line with our expectations. Due to the more balanced number of relevant duplicates between training and test, the models obtain rewards multiple times for the same document resulting in better performance in an evaluation scenario where duplicates are indeed relevant. The results support our hypothesis of the adverse effects unmitigated duplication has on LTR models with small to medium effect sizes. Not all reported results meet the significance level of $p \leq 0.05$, which may be due to the relatively small dataset or because duplication does not confuse the models entirely.

4.2 Bias Towards Duplicate Documents

As an additional and more in-depth case study, we measure whether varying the number of Wikipedia duplicates in the training set can bias models specifically towards ranking Wikipedia documents higher. This evaluation is important, since nDCG@20 alone cannot attest an unbiased model and the vulnerability of a model to duplicates in this way opens up a number of opportunities for SEO abuse. The right-hand column group of Table 2 reports the mean first rank of irrelevant Wikipedia documents in the test sets. Unbiased models should rank irrelevant documents at the rearmost positions, whereas models that rank irrelevant documents at high positions, are clearly biased towards those documents.

We find that all models do indeed rank irrelevant Wikipedia documents significantly higher than the BM25@body baseline. This bias is concerning, although in the case of Wikipedia not entirely unreasonable; Wikipedia documents have an inherently higher probability of being relevant for a topic than other documents (26 % vs. 22 %). Coordinate Ascent seems particularly vulnerable: Training with full redundancy on our worst-case arrangement produces models that rank irrelevant Wikipedia documents within the top 20 (mean first rank: 19), which is significantly mitigated with large effect sizes in the 0 % setting in which duplicates are removed from training (rank: 43, $d = 0.6$) or when the novelty principle *NOV* is applied (rank: 48, $d = 0.8$). Almost all significant changes in Table 2 indicate that deduplication increases the position of the first irrelevant Wikipedia document in the average case, LambdaMart being the only exception. Given its very close fit of the training data, we assume that LambdaMart learns the high relevance probability of Wikipedia documents even without duplicates. In conclusion, our experiments show that duplicates in the training data of LTR algorithms can bias trained models towards these duplicate documents.

5 CONCLUSION AND FUTURE WORK

Our study finds that near-duplicates in web crawls harm retrieval performance, but are unaccounted for in the loss-minimization for learning to rank and in subsequent evaluations. By conducting experiments using 42 features and an array of popular algorithms, we could show that LTR pipelines benefit from removing duplicate documents from the data prior to training the model.

At this point, our observations are limited to the ClueWeb09 corpus and only some popular algorithms. Still, the same observations can most likely be obtained from other corpora and models and we believe that our work provides sufficient motivation to start further exploration of sampling biases in machine-learned ranking. An important line of future work in this direction includes deep-learned LTR models like BERT [7] for document ranking. Another future direction is a more in-depth analysis of the limits of duplication bias on individual documents in the LETOR datasets by creating synthetic duplicates of these documents as an extreme case study.

REFERENCES

[1] Q. Ai, J. Mao, Y. Liu, W. B. Croft. 2018. Unbiased Learning to Rank: Theory and Practice. In *Proc. of CIKM 2018.* ACM, 2305–2306.
[2] Y. Bernstein, J. Zobel. 2005. Redundant Documents and Search Effectiveness. In *Proc. of CIKM 2005.* ACM, 736–743.
[3] O. Chapelle, Y. Chang. 2011. Yahoo! Learning to Rank Challenge Overview. In *Proc. of the Yahoo! Learning to Rank Challenge, held at ICML 2010.* PMLR, 1–24.
[4] C. L. A. Clarke, N. Craswell, I. Soboroff. 2004. Overview of the TREC 2004 Terabyte Track. In *Proc. of TREC.* NIST.
[5] G. V. Cormack, M. D. Smucker, C. L. A. Clarke. 2011. Efficient and Effective Spam Filtering and Re-ranking for Large Web Datasets. *Inf. Retr.* 14, 5 (2011), 441–465.
[6] J. S. Culpepper, F. Diaz, M. D. Smucker. 2018. Research Frontiers in Information Retrieval: Report from the Third Strategic Workshop on Information Retrieval in Lorne (SWIRL 2018). *SIGIR Forum* 52, 1 (2018), 34–90.
[7] Z. Dai, J. Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. In *Proc. of SIGIR 2019.* 985–988.
[8] V. Dang. 2013. The Lemur Project-Wiki-RankLib. *Lemur Project* (2013). Available: https://sourceforge.net/p/lemur/wiki/RankLib.
[9] D. Fetterly, M. Manasse, M. Najork. 2003. On the Evolution of Clusters of Near-Duplicate Web Pages. In *Proc. of LA-WEB 2003.* IEEE, 37–45.
[10] Y. Freund, R. D. Iyer, R. E. Schapire, Y. Singer. 2003. An Efficient Boosting Algorithm for Combining Preferences. *J. Mach. Learn. Res.* 4 (2003), 933–969.
[11] M. Fröbe, J. P. Bittner, M. Potthast, M. Hagen. 2020. The Effect of Content-Equivalent Near-Duplicates on the Evaluation of Search Engines. In *Proc. of ECIR 2020.* Springer, 12–19.
[12] D. Hiemstra, C. Hauff. 2010. MIREX: MapReduce Information Retrieval Experiments. Tech. Report TR-CTIT-10-15. (2010).
[13] M. Ibrahim, M. J. Carman. 2014. Undersampling Techniques to Re-balance Training Data for Large Scale Learning-to-Rank. In *Proc. of AIRS 2014.* Springer, 444–457.
[14] T. Joachims, A. Swaminathan, T. Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proc. of WSDM 2017.* ACM, 781–789.
[15] C. Macdonald, R. L. T. Santos, I. Ounis. 2012. On the Usefulness of Query Features for Learning to Rank. In *Proc. of CIKM 2012.* 2559–2562.
[16] C. Macdonald, N. Tonellotto, I. Ounis. 2012. Effect of Dynamic Pruning Safety on Learning to Rank Effectiveness. In *Proc. of SIGIR 2012.* ACM, 1051–1052.
[17] D. Metzler, W. B. Croft. 2007. Linear feature-based models for information retrieval. *Inf. Retr.* 10, 3 (2007), 257–274.
[18] T. Minka, S. Robertson. 2008. Selection Bias in the LETOR Datasets. In *SIGIR Workshop on Learning to Rank for Information Retrieval.* ACM, 48–51.
[19] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, L. Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proc. of the Workshop on Cognitive Computation at NIPS 2016.* CEUR-WS.org.
[20] Z. Ovaisi, R. Ahsan, Y. Zhang, K. Vasilaky, E. Zheleva. 2020. Correcting for Selection Bias in Learning-to-rank Systems. In *Proc. of WWW 2020.* ACM, 1863-1873.
[21] T. Qin, T. Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013).
[22] P. Serdyukov, N. Craswell, G. Dupret. 2012. WSCD 2012: Workshop on Web Search Click Data 2012. In *Proc. of WSDM 2012.* ACM, 771–772.
[23] G. Vandewiele, I. Dehaene, G. Kovács, L. Sterckx, O. Janssens, F. Ongenae, F. De Backere, F. De Turck, K. Roelens, J. Decruyenaere, S. Van Hoecke, T. Demeester. 2020. Overly Optimistic Prediction Results on Imbalanced Data: Flaws and Benefits of Applying Over-sampling. *CoRR* abs/2001.06296 (2020).
[24] L. Vaughan, M. Thelwall. 2004. Search Engine Coverage Bias: Evidence and Possible Causes. *Inf. Process. Manage.* 40, 4 (2004), 693–707.
[25] X. Wang, M. Bendersky, D. Metzler, M. Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proc. of SIGIR 2016.* ACM, 115–124.
[26] X. Wang, N. Golbandi, M. Bendersky, D. Metzler, M. Najork. 2018. Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proc. of WSDM 2018.* ACM, 610–618.
[27] Q. Wu, C. J. C. Burges, K. M. Svore, J. Gao. 2010. Adapting Boosting for Information Retrieval Measures. *Inf. Retr.* 13, 3 (2010), 254–270.
[28] P. Yang, H. Fang, J. Lin. 2017. Anserini: Enabling the Use of Lucene for Information Retrieval Research. In *Proc. of SIGIR 2017.* ACM, 1253–1256.
[29] B. Zadrozny. 2004. Learning and Evaluating Classifiers under Sample Selection Bias. In *Proc. of ICML 2004.* ACM.
[30] Y. Zhu, Y. Lan, J. Guo, X. Cheng, S. Niu. 2014. Learning for Search Result Diversification. In *Proc. of SIGIR 2014.* ACM, Gold Coast, Australia, 293–302.