Implicit Feedback for Dense Passage Retrieval: A Counterfactual Approach

Shengyao Zhuang
The University of Queensland
Brisbane, QLD, Australia
s.zhuang@uq.edu.au

Hang Li
The University of Queensland
Brisbane, QLD, Australia
hang.li@uq.edu.au

Guido Zuccon
The University of Queensland
Brisbane, QLD, Australia
g.zuccon@uq.edu.au

ABSTRACT

In this paper we study how to effectively exploit implicit feedback in Dense Retrievers (DRs). We consider the specific case in which click data from a historic click log is available as implicit feedback. We then exploit such historic implicit interactions to improve the effectiveness of a DR. A key challenge that we study is the effect that biases in the click signal, such as position bias, have on the DRs. To overcome the problems associated with the presence of such bias, we propose the Counterfactual Rocchio (CoRocchio) algorithm for exploiting implicit feedback in Dense Retrievers. We demonstrate both theoretically and empirically that dense query representations learnt with CoRocchio are unbiased with respect to position bias and lead to higher retrieval effectiveness.

CCS CONCEPTS

• **Information systems** → **Retrieval models and ranking**; **Information retrieval query processing**.

KEYWORDS

Dense passage retrieval, Implicit feedback, Counterfactual Learning

ACM Reference Format:
Shengyao Zhuang, Hang Li, and Guido Zuccon. 2022. Implicit Feedback for Dense Passage Retrieval: A Counterfactual Approach. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22, July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/xxxxxx

1 INTRODUCTION

Dense Retrievers (DRs) are retrieval and ranking approaches where a transformer-based deep language model (e.g., BERT [21]) is used to separately encode queries and documents in low dimensional embeddings (dense vectors), which are then used as representation for matching via vector similarity search (inner product). Although generally less effective than cross-encoder approaches such as mono-BERT [32], DRs offer much lower query latency, overcoming the main barriers to adoption of cross-encoder methods (high query latency and high computational cost).

Dense retrievers are static, in the sense that after a DR has been trained, its encoding mechanism does not change, and so does not its retrieval mechanism. In this paper, we are interested to investigate the use of implicit feedback that is collected by a search engine, and in particular click-through data [4, 18], in the context of DRs. In particular, we are interested to explore how the click signal collected in historical click logs could be used to improve the effectiveness of DRs.

The key idea we put forward in this paper is to adapt current pseudo relevance feedback (PRF) methods for DRs [24] to deal with implicit feedback in terms of clicks. A first sight, this can be achieved by simply removing the PRF assumption that the top-k search results are relevant and therefore form the relevance signal. This assumption can be replaced by the assumption that the clicked search results are what constitutes the relevance signals and can be used to enrich the query representation. This intuition is depicted in Figure 1 where the dense representation of the query $\vec{q}$ is aggregated with the dense representation of the clicked passages $\vec{p_1}$ and $\vec{p_3}$, to create the updated query $\vec{q}'$. Note that a difference exists here between the traditional use of PRF and our adaptation, besides from the type of relevance signal. PRF is commonly used to execute a second round of retrieval, following on from an initial retrieval round without feedback. The methods we put forward in this paper instead tackle the first round of retrieval: the click feedback does not come from the interaction between the current user and the SERP, but from the historic click feedback the system has collected for past queries.

The adaptation of current PRF methods for DRs to dealing with the click signal, however, presents three key challenges.

Challenge 1 (datasets): the training of DRs requires large datasets, such as MS MARCO, that contain both textual passages and relevance labels. These datasets however do not have corresponding click data to be used as implicit feedback to support our investigation[1]. Then, how can existing large datasets for training and evaluating DRs be extended with click information?

To address this challenge, we use the evaluation practice developed in online learning to rank (LTR) and counterfactual LTR, where click models are used to simulate user interactions. This allows us to simulate clicks on passage ranking collections used to study dense retrievers.

Challenge 2 (click bias): it is well known that the click signal presents a number of biases, such as position bias [19, 45]. Then, what is the impact of this bias on the effectiveness of relevance feedback methods for DRs when clicks are used as relevance signal?

[1]While there exist a clicklog that is associated to the MS MARCO dataset (ORCAS [7]) this does not fit the needs for our study – see section 2 for details.

SIGIR '22, July 11–15, 2021, Madrid, Spain

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/xxxxxx

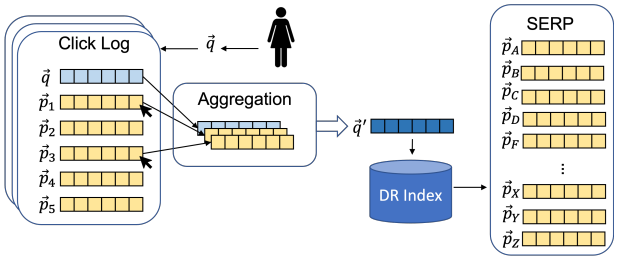


Figure 1: High level visualisation of our proposed method for exploiting implicit feedback from historic click logs to enhance dense retrievers (DRs).

How can this bias be removed from the click signal – and does this improve search effectiveness?

To investigate and address this challenge, we first study the impact of implicit feedback on the effectiveness of DRs under different user conditions and click bias. We then propose a counterfactual method for DRs that mitigates the impact of bias in the click signal. Specifically, our method relies on a historic click log to estimate the likelihood of position bias associated to a query $\vec{q}$. This is then used to adjust the weight of the dense representation associated to a feedback passage $\vec{p}$ when aggregating this signal with that from other passages and the original query $\vec{q}$ to compute the new dense query representation $\vec{q}'$.

Challenge 3 (counterfactual DRs with unseen queries): our counterfactual method requires that, for any given query to be scored, the historic click log contains interactions with SERPs for that query. While this is a reasonable expectation for popular queries, it is likewise reasonable to expect that at times users will enter queries that are not present in the historic click log. What strategy can then be used to adapt the proposed counterfactual method to cases in which the query has not been previously observed in the historic click log?

To address this final challenge, we first identify a strategy to generate new queries not in, but yet related to[2], the historic query log we create as dataset. We then devise a simple but effective strategy to identify the portion of the signal from the historic click log to be used by our counterfactual method for the current query.

In summary, this paper explores a novel research direction: that of exploiting implicit signals such those in historic click logs to enhance DRs. With this respect, we show that evaluation practices developed for online and counterfactual LTR can be adopted to this context. In addition, we show that the historic click signal can be highly informative for DRs, making these methods highly effective. On the other hand, we show that the noise and bias typically found in click-through data have a significant detrimental effect. To address this issue we contribute a novel counterfactual method for DRs to exploit biased historic click signals, called Counterfactual Rocchio (CoRocchio) and investigate the key factors that influence its effectiveness, including its application to two different DRs and how it can deal with queries not present in the historic click log (unseen queries).

[2]Such that the signals from the historic click log bear a valuable signal for the unseen queries.

2 RELATED WORK

This paper relates to the creation of effective dense retrievers methods. *Dense Retrievers* (DRs) [14, 20, 22, 27, 42] utilise a BERT-based (or variation of) dual-encoder architecture to separately encode queries and passages into a shared embedding space. DRs have shown to have high effectiveness paired with low query latency across differenrent retrieval tasks and datasets. However, current research directions in the DRs space focus on the development of effective training methods, especially related to negative sampling [11, 23, 42], and the integration of pseudo relevance feedback mechanisms [24, 43]. The exploitation in the context of DRs of implicit relevance signals such as clicks collected in a search engine click log has so far received no attention. Our paper aims to fill this gap and lay the foundations for this new line of research for DRs.

The development of effective techniques for the integration of PRF in DRs is relevant to our work from two angles. First, both that work and ours attempt to improve the query representation by exploiting signals in addition to those provided by the query alone, though the signal we use is different from that considered by PRF. Second, the way in which we integrate the historic click signal into DRs is fundamentally similar, in its more basic instantiation, to the vector PRF methods for DRs devised by Li et al. [24]. Specifically, they investigated two simple instantiations, Average and Rocchio, that perform PRF with DRs in a zero-shot manner, i.e. without requiring the training of a new neural model or fine tuning of an existing one. Alternative, more complex approaches for integrating PRF into DRs exist. For example, Yu et al. [43] replaced the query encoder in ANCE [42] by purpously training a new PRF query encoder. This new PRF query encoder concatenates the original query text and the text from the PRF passages to form a new query, which is then encoded and used to perform matching. A fundamental difference however exists between PRF methods for DRs and the methods we propose in this paper, aside from the type of relevance signal that they consider. PRF methods are commonly part of a two-stage pipeline: a first round of retrieval is performed using the original query; this is then used as relevance signal to perform a second round of retrieval. The use of this second stage adds to the overall query latency. Our methods instead are single-stage: they directly modify the query representation exploiting the click signals from previous queries and retrieve using such modified representation. Thus, the methods we propose here have minimal or no effect on the query latency of the underlying DRs.

In investigating how to exploit historic click signals for improving DRs, we make two key connections to the area of online and counterfactual LTR [16, 19, 36, 38, 48]. First, we rely on some of the components of the standard evaluation practice in that area, such as the use of click models [6, 13] to simulate clicks and therefore generate the historic click log [16, 17, 19, 37]. Borrowing methods from this previous research allows us to have the foundational tools to address Challenge 1, outlined in section 1. Second, to address the presence of bias in the click signal (Challenge 2 in section 1), we again tap into the research in counterfactual LTR, and integrate the notion of counterfactual de-biasing in the proposed Rocchio mechanism for DRs. Our work bears a resemblance of tabular models in LTR, where a ranker is learnt specifically for an individual query from direct interactions with the user, i.e., they model each query as

an independent ranking problem, but therefore cannot generalise beyond that query [37, 50]. Despite these connections, we note that online/counterfactual LTR methods themselves are not comparable to DRs (thus of course are not considered as baseline for evaluation). In fact, DRs may be part of a more complex pipeline that involve such LTR approaches, and perhaps they can even be used as one of the many features LTR methods exploit.

One of the challenges we had to address in our work was the absence of a dataset adequate to train and evaluate DRs, like TREC DL, and that also contained historic click logs. We note that the MS MARCO dataset comes with an associated click log: ORCAS [7]. However, this click log refers to the document part of MS MARCO, and thus it is difficult to derive from it clicks that are associated to the passages in the MS MARCO and TREC DL datasets that are typically used for researching DRs on the passage ranking task. We note that only a subset of the documents in MS MARCO has a mapping to the passage part of the dataset. Another limitation of ORCAS is that clicks are recorded as query-document pairs and no information regarding the rank position the document was displayed at in the SERP has been recorded for that click. This affects our ability to derive position bias information. Hence we cannot use ORCAS for our experiments.

3 METHOD

Next we introduce our counterfactual Rocchio (CoRocchio) method which exploits the click signal from an historic query log to compute a new dense representation of the user query. CoRocchio relies on the framework of dense retrievers: queries and passages are encoded using a dense pre-trained deep language model encoder (e.g. ANCE, DPR, etc.) and retrieval is performed by computing the inner product between the dense representation of the query and that of the passages. In CoRocchio, however, the query representation is computed by aggregating the dense representation of the user's query with that of passages in the historic query log that have been clicked for that query (Figure 1).

CoRocchio relies on two key intuitions: (1) The passages that have received a click in the query log are often the user-preferred passages for that query; (2) Aggregating the dense representation of the query with these of the user-preferred passages will result in a new query representation that better matches user preference.

The first intuition has been shown valid in previous work on learning to rank [17], online learning to rank and evaluation [35, 36, 44, 48] and counterfactual learning to rank [19, 37].

The second intuition is also convincing: the correct aggregation of the original dense representation of the query with these of the user-preferred passages will move the query representation towards those of the preferred passages. Thus, other passages that have a similar dense representation will obtain a larger score with the new query representation than the old one – and these passages are highly likely to also be preferred by the users.

3.1 DRs with Implicit Feedback

Relying on the above intuitions, we first define the Rocchio algorithm with DRs for exploiting historic clicks. This method is an adaptation of Li et al.'s Rocchio PRF for DRs [24] to the situation in which pseudo relevance feedback is replaced by the implicit feedback derived from the click-through signal.

Let $c(p_i) \in \{0, 1\}$ be the binary function indicating if a user clicked on the passage p_i ($c(p_i) = 1$) or not ($c(p_i) = 0$). Furthermore, let $\vec{q}$ and $\vec{p}$ be the query q dense representation and passage p dense representation, respectively. Each time the query is encountered in the query log, a ranking r_q of passages has been generated by the baseline DR and clicks collected. The set of all historic rankings for query q is indicated with R_q.

Now, let's consider the case in which query q is issued to the online system and the historic query log wants to be used to improve the effectiveness of the baseline DR ranker. To do so, we use the Rocchio formula below to create a new query dense representation $\vec{q}'$ from click information:

$$\begin{aligned}\vec{q}' &= \frac{1}{|R_q|} \sum_{r_q \in R_q} \left[\alpha \cdot \vec{q} + \beta \cdot \sum_{p_i \in r_q} \vec{p}_i \cdot c(p_i) \right] \\ &= \alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r_q \in R_q} \sum_{p_i \in r_q} \vec{p}_i \cdot c(p_i)\end{aligned} \tag{1}$$

where α and β are the Rocchio parameters: α controls the weight assigned to the original query and β controls the weight assigned to the aggregated dense representation from the clicked passages. This new query representation $\vec{q}'$ is then used to rank passages according to the inner product between $\vec{q}'$ and the passages' dense representations.

In the edge case when all the relevant passages for a query have been displayed in the SERPs in the historic logs and users have clicked on all relevant passages, and have not clicked any not relevant ones, then new query representation $\vec{q}^*_c$ only aggregates relevant passages:

$$\vec{q}^* = \alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r \in R_q} \sum_{p_i \in r_q} \vec{p}_i \cdot y(p_i) \tag{2}$$

where $y(p_i) = 1$ if p_i is relevant or $y(p_i) = 0$ otherwise.

3.2 CoRocchio: Unbiased Aggregation

Equation 1 treats every click equally. However, the click signal presents a number of biases, the strongest being position bias [19, 45]. Position bias manifests as the tendency of the users to click more often (or more likely) on passages that are ranked earlier in the SERP. Not accounting for position bias, as it is done in Equation 1, may lead to suboptimal effectiveness.

Next, we mathematically prove that the Rocchio method for DRs of Equation 1 is biased with respect to position and thus leads to suboptimal results. Following previous work in counterfactual learning to rank [19, 37], we develop this demonstration in the restrictive setting that clicks occur only and only if a passage is relevant and examined by the user – extension of this proof to nosier conditions is possible (though not trivial) but is out of scope for this paper.

Let assume that the probability of a passage p_i to be examined by a user (known as propensity) only depends on its rank position in the SERP and that the user will click on every relevant passage they examined (i.e., no click noise). Thus, the probability of a click on d_i is:

$$P\left(c(p_i)\right) = P(o_i) \cdot y(p_i) \tag{3}$$

where $P(o_i)$ is the examination propensity of rank position i. If we treat clicks as an unbiased relevance signal, then the expectation of $\vec{q}$ is:

$$\begin{aligned}\mathbb{E}_o\left[\vec{q}'\right] &= \mathbb{E}_o\left[\alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r_q \in R_q} \sum_{p_i \in r_q} \vec{p}_i \cdot c(p_i)\right] \\ &= \alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r_q \in R_q} \sum_{p_i \in r_q} \vec{p}_i \cdot P(o_i) \cdot y(p_i) \neq \vec{q}^*\end{aligned} \tag{4}$$

This means $\vec{q}'$ is a biased estimate of $\vec{q}^*$. According to empirically collected data [12], higher rank positions usually have higher examination probability. Thus, $\vec{q}'$ biasedly assigns higher weights to clicked passages displayed at earlier ranks in the SERP.

To overcome the effect of position bias in the click log, inspired by counterfactual learning to rank practice, we propose the Counterfactual Rocchio (CoRocchio) which uses the inverse propensity scoring (IPS) [15, 19] to de-bias the click signal:

$$\text{CoRocchio}\left(\vec{q}, P(o)\right) = \alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r_q \in R_q} \sum_{p_i \in r_q} \frac{\vec{p}_i}{P(o_i)} \cdot c(p_i) \tag{5}$$

CoRocchio generates an unbaised estimate of $\vec{q}^*$ if every relevant passage ($y(p_i) = 1$) has a positive examination propensity ($P(o_i) > 0$):

$$\begin{aligned}&\mathbb{E}_o\left[\text{CoRocchio}\left(\vec{q}', P(o)\right)\right] \\ &= \mathbb{E}_o\left[\alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r_q \in R_q} \sum_{p_i \in r_q} \frac{\vec{p}_i}{P(o_i)} \cdot c(p_i)\right] \\ &= \alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r_q \in R_q} \sum_{p_i \in r_q} \frac{\vec{p}_i \cdot P(o_i)}{P(o_i)} \cdot y(p_i) \\ &= \alpha \cdot \vec{q} + \frac{\beta}{|R_q|} \cdot \sum_{r_q \in R_q} \sum_{p_i \in r_q} \vec{p}_i \cdot y(p_i) = \vec{q}^*\end{aligned} \tag{6}$$

Intuitively, CoRocchio assigns more weights to clicked passages that appear at lower ranks in the SERP by dividing their examination propensity, hence discounting the position bias. This requires to know the propensity $P(o)$ a priori. Following standard counterfactual learning to rank experimental settings [19] and for simplicity, we assume in our experiments the propensities are known and regard propensity estimation as being beyond the scope of this paper. However, we note propensity estimation is a well-studied area on its own and many recent studies have considered estimating such propensities from historical click-logs [2, 3, 10].

3.3 Dealing with Unseen Queries in CoRocchio

Above, we have shown CoRocchio can unbiasedly estimate the optimal query representation for dense retriever from an historic click log. One drawback of CoRocchio is that it can only be used for queries that appear in the historic click log, resembling a tabular-based ranker [37, 50] (at the next of not being LTR-based): it cannot deal with unseen queries. This is acceptable for popular queries: they would be logged many times by the search engine. However, the majority of queries logged by search engines are tail queries or have never been logged before [40].

In order to generalise CoRocchio to unseen queries, we devise the CoRocchio with Approximate Nearest Neighbor Query (CoRocchio-ANN) algorithm. CoRocchio-ANN leverages the property of DR encoders that similar queries would have similar dense representation encodings. More specifically, we use the baseline DR encoder to encode the unseen query into its dense representation $\vec{q_u}$. Then we use $\vec{q_u}$ to search for the top-k nearest neighbor queries in the historic query log. Let Q be the query set of top-k nearest neighbor ($|Q| = k$); then the new query vector for $\vec{q_u}$ is:

$$\begin{aligned}&\text{CoRocchio-ANN}(\vec{q_u}, P(o)) \\ &= \alpha \cdot \vec{q_u} + \frac{\beta}{|Q| \cdot |R_q|} \cdot \sum_{q \in Q} \sum_{r_q \in R_q} \sum_{p_i \in r_q} \frac{\vec{p}_i}{P(o_i)} \cdot c(p_i)\end{aligned} \tag{7}$$

In other words, we use the average unbiased passage representation aggregation for the top-k nearest neighbor queries to compute the new query representation for the unseen query.

4 EXPERIMENTAL SETTINGS

To study the impact implicit feedback in terms of click signal has on the effectiveness of DRs and the effectiveness of our proposed CoRocchio, we devise a number of empirical experiments aimed at investigating: (1) the effectiveness of CoRocchio when using the signal from the historic click log (biased and unbiased) for queries present in the log (results in section 5.1), and (2) the effectiveness of CoRocchio-ANN when exploiting the historic click signal to answer previously unseen queries (results in section 5.2)

4.1 Dataset

We use the TREC Deep Learning Track Passage Retrieval Task 2019 [8] (DL 2019) and 2020 [9] (DL 2020). DL2019 and DL2020 contain 43 and 54 judged queries each. The relevance judgements for both datasets range from 0 (not relevant) to 3 (highly relevant); however, relevance label 1 indicates passages on-topic but not relevant and hence we relabel these passages to 0 when we compute binary relevance metrics (e.g., MAP, Recall). The passage collection is the same as the MS MARCO passage ranking dataset [31], a benchmark English dataset for ad-hoc retrieval tasks with around 8.8 million passages. The difference between TREC DL and MS MARCO is that queries in TREC DL have several judgements per query (215.3/210.9 on average for 2019/2020), instead of only an average of one judgement per query for MS MARCO.

We do not use MS MARCO dataset to evaluate methods in this work. This is because the number of judgments per query is crucial in our experiments since we follow the standard unbiased LTR practice, which relies on relevance judgments and a click model (more details in section 4.2) to synthetically generate user clicks. Since MS MARCO contains only one judgment per query, if used in this context the click model will generate extremely biased click data, largely differing from real-world user behaviour.

Table 1: Click probabilities for different user behaviours.

	$P(c(p)=1\|o=1,y(p))$			
$y(p)$	0	1	2	3
perfect	0.0	0.0	1.0	1.0
noisy	0.2	0.4	0.8	0.9

4.2 Synthetic User Behavior

To fully control users' biases and noise so that algorithms can be tested under different, controllable conditions, it is common for research in online LTR and counterfactual LTR to simulate users' clicks by relying on the relevance labels recorded in offline datasets [16, 19, 36, 38, 47, 48]. Following this practice, we use a click model to simulate click behaviour and create a synthetic click log.

In our experiments, clicks are simulated based on two fixed variables: the click probability and the position bias. The click probability $P(c(p=1)|o=1,y(p))$ is the probability of a user clicking on a passage after exam it. This probability is conditioned on the passage's relevance label $y(p)$. We set two types of click behaviour: *perfect* and *noisy*. The click probability of the *perfect* click behaviour is proportional to the relevance level of the passages, and has 0 probability for non-relevant passages. This simulates an ideal user that is able to always determine the relevance of a passage in the SERP. The *noisy* click behaviour mimics instead a realist behaviour on SERPs by assigning a small click probability to non-relevant passages and a small skip probability to relevant passages. Table 1 provides the click probabilities for the two user models.

Position bias is modelled by the passage observation probabilities $P(o_i=1)$; we assume the observation probabilities only depend on the rank position of the passage and set these probabilities as:

$$P(o_i=1)=\left(\frac{1}{i}\right)^{\eta} \tag{8}$$

where i is the rank position of the given passage d and η is a parameter that determines the level of position bias. Following Joachims et al. [19] and Jagerman et al. [16], we set $\eta=1$ for our main experiments, while we investigate the impact of different η values in Section 5.1. Thus, the probability of a click occurring on a passage at rank k in the result list is:

$$P(c(p)=1)=P(c(p)=1|o_i=1,y(p))\cdot P(o_i=1) \tag{9}$$

With the click model, we then create an historic click log by issuing each query in the TREC DL datasets to the DRs and record the top 10 retrieved passages and the simulated clicks on these passages. We repeat this simulation 1,000 times for each query to log enough click signal; although we note that a lower or higher number of simulations provide similar trends and observations to the ones reported here.

4.3 Dataset Augmentation with Synthetic Query Generation

To evaluate CoRocchio-ANN, which allows to use DRs with implicit feedback in presence of queries not seen in the historic query log, a dataset with unseen queries that are related to those that have been observed in the log is required; and these queries need also to have relevance judgements. Because TREC DL has only a small set of judged queries, and they are unrelated to each other, withholding a subset of TREC DL queries is as unseen queries is not possible. We then have to devise a different avenue to generate a dataset that allows to study this aspect of CoRocchio. To this aim, we adapt the docTquery-T5 method [33] to augment the current TREC DL datasets with unseen, but related queries with associated relevance judgements.

Table 2: Examples of original queries in DL2019 (the first two rows) and DL2020 (the second two rows) vs the queries generated from one of their relevant passages by docTquery-T5.

Original queries	Generated queries
do goldfish grow	how big do shubunkin fish get
what is wifi vs bluetooth	what is the difference between bluetooth and wifi?
who is aziz hashim	who is hashim franchisor
do google docs auto save	how do you save google docs updates

The docTquery-T5 method is a T5 language model [39] that is fine-tuned on the task of generating relevant queries for any given passage. It has been shown effective at generating high quality queries for the task of corpus summarisation [41] and passage expansion [25, 29, 49]. We use docTquery-T5 to generate a query from each passage that has been judged relevant [3] for one of the original queries in the TREC DL dataset. We assume that the passages relevant to the original query for which a passage was taken for query generation, are also relevant to the generated query. With docTquery-T5, it may be possible that the generated query and the original query are identical: in this case we sample again a query from docTquery-T5 to ensure the query generated from a passage is different from the original query for that passage. Since we generate a query for each relevant passage given an original TREC DL query, on average we generate 58 synthetic queries per each original query in TREC DL 2019 and 31 for TREC DL 2020. In total, we generated 2,501 extra queries for DL2019 and 1,666 extra queries for DL2020. In our experiments, we split the augmented query set into a seen query set (80% of the generated queryes) and an unseen query set (20%). The seen query set is used to simulate the historic click log while the unseen query set is used for evaluating CoRocchio-ANN.

Table 2 reports examples of original and generated queries. As expected, the generated queries are similar to the original query. While there are cases in which the original and generated queries differ enough that some of the documents relevant to the original query may not be relevant to the generated query, we empirically observed these to not be frequent enough to be a main source of error in the evaluation – we note that even in a high quality collection such as TREC DL relevance assessments are at time noisy [5].

4.4 Baselines and Metrics

We are the first to study how to exploit implicit feedback, in particular the click-through signal, with DRs – thus baselines are limited.

[3] We treat passages with relevance levels 2 and 3 as relevant passages and exclude those with label 1.

Table 3: Logged query results. † indicates statistically significant differences (p=0.05) between DRs with click signal vs. their respective DRs with PRF signal. Differences between Rocchio vs. CoRocchio are marked with ⋆ if statistically significant.

	TREC DL2019				TREC DL2020			
Method	**nDCG@10**	**nDCG@100**	**Recall@1000**	**MAP**	**nDCG@10**	**nDCG@100**	**Recall@1000**	**MAP**
BM25	0.4973	0.4981	0.7450	0.2901	0.4876	0.4914	0.8031	0.2876
ANCE [42]	0.6452	0.5540	0.7554	0.3710	0.6458	0.5679	0.7764	0.4076
TCT-ColBERTv2 [28]	0.7204	0.6318	0.8261	0.4469	0.6882	0.6206	0.8429	0.4754
Pseudo-relevance feedback								
BM25 + RM3 [1]	0.5231	0.5263	0.7792	0.3377	0.4808	0.5145	0.8286	0.3056
ANCE + VPRF [24]	0.6561	0.6018	0.7562	0.4278	0.6221	0.5636	0.7875	0.4098
ANCE-PRF [43]	0.6807	0.5950	0.7912	0.4253	0.6948	0.5947	0.8148	0.4452
TCT-ColBERTv2 + VPRF [24]	0.6982	0.6556	0.8633	0.4797	0.6678	0.6058	0.8514	0.4697
Exploits Implicit feedback								
Perfect and unbiased user								
ANCE + Rocchio	0.7543†	0.6557†	0.8144	0.5045†	0.7438†	0.6227	0.8170	0.5077
TCT-ColBERTv2 + Rocchio	0.7963†	0.7233	0.9001	0.5778†	0.8095†	0.6998†	0.9033†	0.5935†
Perfect and biased user								
ANCE + Rocchio	0.7243	0.6393†	0.8047	0.4764	0.7199	0.6140	0.8164	0.4848
ANCE + CoRocchio	0.7522†⋆	0.6558†⋆	0.8149⋆	0.5033†⋆	0.7413†⋆	0.6227	0.8175	0.5080
TCT-ColBERTv2 + Rocchio	0.7883†	0.7121†	0.8948	0.5592†	0.7893†	0.6846†	0.8970	0.5571†
TCT-ColBERTv2 + CoRocchio	0.7963†	0.7237†	0.9004	0.5774†	0.8093†⋆	0.6996†⋆	0.9014†	0.5935†
Noise and unbiased user								
ANCE + Rocchio	0.7167	0.6314†	0.7848	0.4665	0.6981	0.6057	0.8070	0.4698
TCT-ColBERTv2 + Rocchio	0.7759†	0.7062†	0.8813	0.5495†	0.7559†	0.6647†	0.8767†	0.5424†
Noise and biased user								
ANCE + Rocchio	0.7019	0.6218	0.7921	0.4589	0.6877	0.6044	0.8114	0.4593
ANCE + CoRocchio	0.7119	0.6295†	0.7836	0.4657	0.6965	0.6049	0.8060	0.4688
TCT-ColBERTv2 + Rocchio	0.7688†	0.7028†	0.8830	0.5381	0.7444†	0.6611†	0.8828†	0.5276†
TCT-ColBERTv2 + CoRocchio	0.7730†	0.7054†	0.8816	0.5503†	0.7568†	0.6662†	0.8827†	0.5446†

We study the use of the proposed CoRocchio method and its variants Rocchio (without counterfactual de-biasing) and CoRocchio-ANN (for unseen queries) in the context of two DRs: ANCE [42] and TCT-ColBERTv2 [28], although the method could be generalised to other DRs. ANCE and TCT-ColBERTv2 then naturally represent two baselines for CoRocchio. We also report the effectiveness of BM25 + RM3, a representative bag-of-words PRF method, since it is often used as a strong bag-of-word baseline in DRs research.

Strictly speaking CoRocchio is not a relevance feedback mechanism of the like of PRF because, at the net of replacing the pseudo relevance signal with the implicit relevance signal, CoRocchio does not consider the relevance feedback from an initial round of retrieval , but the implicit feedback from an historic click log (and thus past searches, possibly performed by other users). Despite this, the CoRocchio aggregation function is similar to that of the vector PRF (VPRF) methods proposed for DRs [24], especially if no counterfactual de-biasing is used. We therefore also use VPRF as a baseline: this method can be applied to any DR. This allows us to establish the value of the implicit relevance signal over the PRF signal in the context of DRs, since the scoring method results to be similar (at the net of the counterfactual de-bias). We also then report as baseline a different way of performing PRF with DRs: the recently proposed ANCE-PRF [43]. Instead of aggregating the dense representations of the feedback passages as in CoRocchio and VPRF, ANCE-PRF trains a new encoder to be able to generate effective representations of the concatenation of the original text of the query and that of the feedback passages. This method is only available for the ANCE DR.

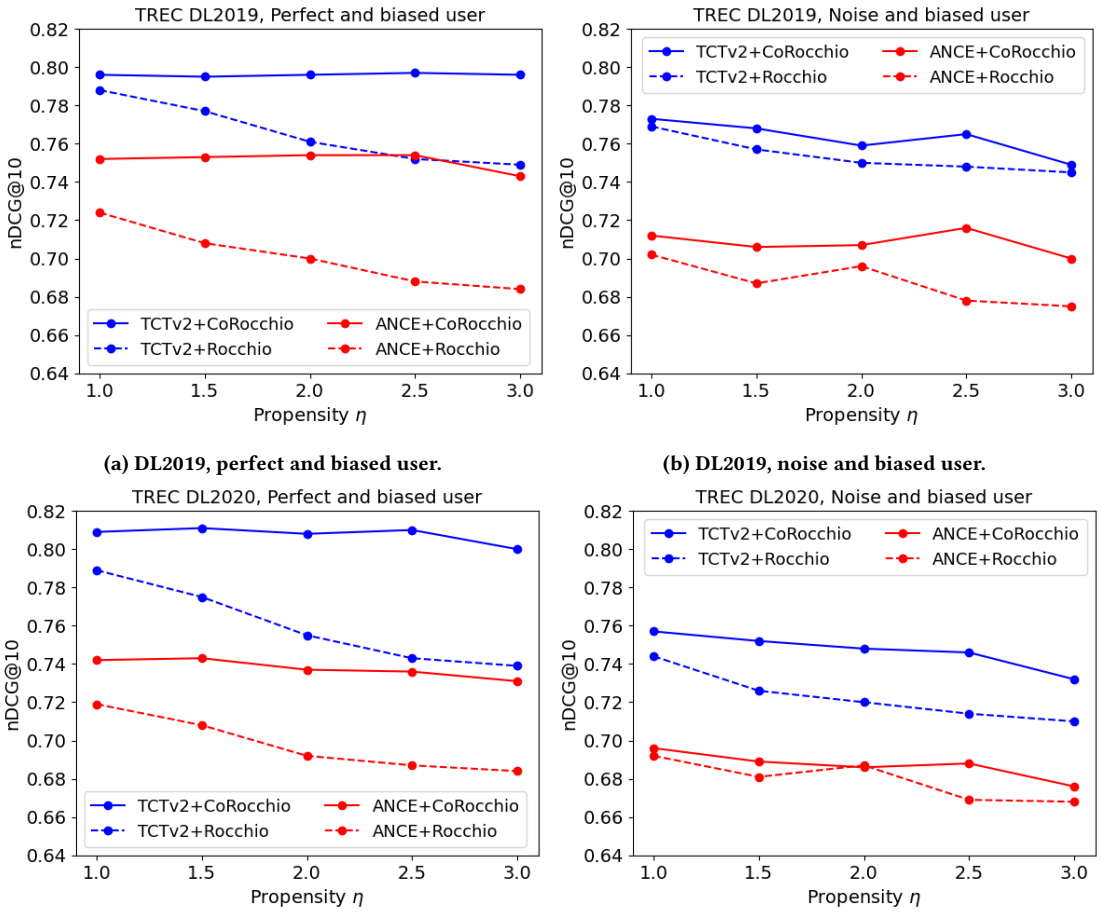
For all Rocchio-based methods (our Rocchio, CoRocchio, CoRocchio-ANN and VPRF), we set $\alpha = 0.4$ and $\beta = 0.6$ for fair comparison. Our method and all baselines are implemented using Pyserini [26] and the code will be made publicly available upon acceptance. We compare the methods with respect to nDCG@10, nDCG@1000, Recall@1000 and MAP, which are commonly used in TREC DL. Statistical differences between methods' results are computed using two-tailed paired t-test with Bonferroni correction.

5 RESULTS

5.1 Implicit Feedback and CoRocchio

We first consider the results obtained when using the proposed method for DRs to exploit the historical click log for queries observed in the log itself (logged queries). The main results for such logged queries are presented in Table 3.

Let us briefly consider the baselines. Comparison between BM25, BM25+RM3, ANCE and TCT-ColBERTv2 reinforce previous findings in the literature: DRs are generally more effective than baseline bag-of-words-methods (including with PRF). Similarly, the use of PRF for DRs generally improves DR baselines for Recall@1000 and MAP; however, for the shallow metrics such as nDCG@10, the improvement of PRF is unstable [24]. For example, ANCE + VPRF and TCT-ColBERTv2 + VPRF have lower nDCG@10 than the respective DRs without PRF; the only PRF method that consistently



(a) DL2019, perfect and biased user.

(b) DL2019, noise and biased user.

(c) DL2020, perfect and biased user.

(d) DL2020, noise and biased user.

Figure 2: nDCG@10 score of ANCE and TCT-ColBERTv2 with Rocchio or CoRocchio under different propensity settings.

outperforms the corresponding DR baseline is ANCE-PRF [43]. We note that VPRF is a zero-shot method (untrained) while ANCE-PRF requires training, so these results are unsurprising.

Next, let us consider the results obtained when exploiting the click logs.

When the click model is that of the perfect user, and no bias is injected (**Perfect and unbiased user**), then the users' clicks recorded in the log have been on all and only the relevant passages present in the SERPs. This means there is no click noise and no position bias ($P(o_i) = 1$ always) and thus CoRocchio reduces to Rocchio (thus we do not report CoRocchio in Table 3 for this setting). Under this setting, exploiting the historic click signal in DRs using our Rocchio returns significant effectiveness boosts across all datasets and metrics for both of the considered DRs. This result is to be expected: in this setting the click signal is a very clear indicator of the relevant passages, despite this is only collected only for a subset of the relevant passages for a query, i.e. those displayed in the SERP (10 passages) displayed in the historic click log, while the methods are evaluated across the full ranking (top 1,000 passages). However, we still believe this edge case result is valuable as it suggests that, for IR applications where the user real preference feedback is available, such as during the screening process in technology assisted reviews [30, 34, 46], DRs can be adapted to exploit such a signal in a very effective manner.

When bias is added, even in the absence of click noise (**Perfect and biased user**), we observe decreases in the effectiveness of Rocchio, regardless of the DR. For example, nDCG@10 on DL2019 for ANCE + Rocchio goes from 0.7543 for the perfect and unbiased user to 0.7243 when bias is introduced; TCT-ColBERTv2 + Rocchio also experiences drops, and drops are indeed observed across all evaluation metrics and datasets. These results indicate that position bias has a negative impact on learning an effective dense query representation by exploiting the click log signal with our Rocchio method. On the other hand, when CoRocchio is used to obtain the new dense query representation that exploits the information in the query log, higher effectiveness is observed. In fact, after de-biasing and aggregating the click signal with CoRocchio, all DRs obtain results that are similar to those obtain in absence of click bias. These results empirically prove what we mathematically demonstrated in section 3.2: that CoRocchio can remove the position bias present

Table 4: Unseen query results. † indicates statistically significant differences (p=0.05) between DRs with click signal vs. their respective DRs with PRF signal. Differences between Rocchio vs. CoRocchio are marked with * if statistically significant.

	TREC DL2019				TREC DL2020			
Method	**nDCG@10**	**nDCG@100**	**Recall@1000**	**MAP**	**nDCG@10**	**nDCG@100**	**Recall@1000**	**MAP**
BM25	0.2996	0.2578	0.4258	0.1409	0.2164	0.2128	0.4557	0.1140
ANCE [42]	0.4024	0.3400	0.4622	0.2156	0.3251	0.2725	0.4802	0.1679
TCT-ColBERTv2 [28]	0.4203	0.3690	0.5211	0.2417	0.3298	0.2888	0.5336	0.1848
Pseudo-relevance feedback								
BM25 + RM3 [1]	0.3302	0.3031	0.4971	0.1850	0.2430	0.2474	0.5124	0.1443
ANCE + VPRF [24]	0.4323	0.3840	0.5084	0.2575	0.3445	0.3071	0.5250	0.2020
ANCE-PRF [43]	0.4297	0.3710	0.5008	0.2440	0.3482	0.3006	0.5201	0.2009
TCT-ColBERTv2 + VPRF [24]	0.4523	0.4142	0.5769	0.2906	0.3411	0.3284	0.5801	0.2267
Exploits Implicit feedback								
Perfect and unbiased user								
ANCE + Rocchio-ANN	0.6404†	0.5155 †	0.6036†	0.3625†	0.5999†	0.4569†	0.6489†	0.3292†
TCT-ColBERTv2 + Rocchio-ANN	0.6904†	0.5588†	0.6752†	0.4059†	0.6314†	0.5091†	0.7289†	0.3918†
Perfect and biased user								
ANCE + Rocchio-ANN	0.5872†	0.4768†	0.5767†	0.3284†	0.5287†	0.4120†	0.6168†	0.2884†
ANCE + CoRocchio-ANN	0.6395†*	0.5152†*	0.6036†*	0.3625†*	0.5986†*	0.4564†*	0.6488†*	0.3288†*
TCT-ColBERTv2 + Rocchio-ANN	0.6325†	0.5242†	0.6490†	0.3747†	0.5698†	0.4645†	0.6992†	0.3451†
TCT-ColBERTv2 + CoRocchio-ANN	0.6897†*	0.5585†*	0.6757†*	0.4061†*	0.6309†*	0.5085†*	0.7288†*	0.3915†*
Noise and unbiased user								
ANCE + Rocchio-ANN	0.5935†	0.4919†	0.5992†	0.3402†	0.5342†	0.4273†	0.6349†	0.2941†
TCT-ColBERTv2 + Rocchio-ANN	0.6323†	0.5324†	0.6686†	0.3815†	0.5635†	0.4801†	0.7218†	0.3517†
Noise and biased user								
ANCE + Rocchio-ANN	0.5635†	0.4671†	0.5788†	0.3174†	0.4996†	0.4028†	0.6157†	0.2730†
ANCE + CoRocchio-ANN	0.5923†*	0.4919†*	0.5979†*	0.3403†*	0.5337†*	0.4274†*	0.6350†*	0.2941†*
TCT-ColBERTv2 + Rocchio-ANN	0.6068†	0.5142†	0.6525†	0.3633†	0.5271†	0.4499†	0.7005†	0.3219†
TCT-ColBERTv2 + CoRocchio-ANN	0.6327†*	0.5325†*	0.6687†*	0.3813†*	0.5630†*	0.4802†*	0.7224†*	0.3515†*

in the historic click log, delivering increased effectiveness for the employed DRs.

Next we investigate the impact of click noise, by considering the noisy click model in place of the perfect. When no click bias is present (**Noise and unbiased user**), Rocchio and CoRocchio result to be identical (as when the setting Perfect and unbiased user was considered). Comparing the results obtained in this setting with those for the perfect user in the unbiased case, we observe significant drops in effectiveness – in fact these drops are larger than those imposed by the introduction of bias in the perfect click model. This suggests that, at least in the settings investigated in our simulations, click noise is more detrimental to DRs effectiveness than click position bias.

This suggests that click noise has bigger negative impact for DRs that exploit historic click log.

On the other hand, when both click noise and position bias are present in the click log (**Noise and biased user**) – the most realistic amongst the settings considered – we observe Rocchio for DRs obtains the lowest effectiveness compared to other settings (but still higher than baselines) across all metrics except Recall@1000. The higher Recall@1000 observed for this setting may be because the noisy clicks that occur at lower-ranked positions have a higher negative contribution to Recall@1000 than the positive contribution coming from correct clicks. When position bias is present, lower-ranked passages have a lower observation probability and thus attract fewer noisy clicks. However, if CoRocchio is used in the Noise and biased user setting, we observe that it can effectively reduce, if not eliminate, the position bias, aligning its result in this context to those obtained for noisy clicks and no bias. Interestingly, also in this case we observe higher Recall@1000 than when noisy but unbiased clicks were used.

Influence of user propensity (η). In the previous experiments, to simulate user position bias on SERPs η was set to 1. In Figure 2, we report the influence of higher values of user position bias η on the effectiveness of Rocchio and CoRocchio. For this, we use different values of η to simulate different user observation probabilities, where higher η means the users are more biased towards the top ranked passages. When $\eta = 3$, users will have less than 5% of chances to observe passages beyond rank 3 (extreme bias). For this experiment, we report the nDCG@10 achieved by different methods on both TREC DL2019 and 2020. From Figure 2, it is clear that our CoRocchio outperforms Rocchio across all datasets, models and η values, empirically demonstrating the benefits of de-biasing position bias. We also note that, under the perfect and biased user setting, the effectiveness of Rocchio decreases with the increase of η. This shows the negative impact of user position bias for DRs that exploit click feedback: the more extreme the bias is, the lower the DRs' effectiveness. On contrary, the effectiveness of CoRocchio is left relatively unchanged as η increases. This means that, if no click noise is present, CoRocchio can correctly remove the user position bias, no matter how extreme this bias is. On the other hand, for the noise and bias click setting, both Rocchio and CoRocchio show lower nDCG@10 when η is larger, but CoRocchio is always better than Rocchio.

In summary, the empirical results analysed in this section suggest that our methods, devised to exploit the implicit feedback signal from historic click logs in the context of DRs, significantly improve the search effectiveness of the considered DRs, regardless of the presence of noise and bias in the click signal.

5.2 Unseen Queries and CoRocchio-ANN

Next we investigate the effectiveness of our proposed CoRocchio-ANN that can deal with queries that have not been observed in the historical click log: this is done by searching the top-k most similar queries in the click log to the current query, and exploit their clicked passages' representation. The ANN mechanism guarantees to find the k-closest logged queries and we find that setting $k = 3$ worked best after manual inspection of a subset of queries in the training data. Hence we fix $k = 3$ throughout this experiment. Note that the ANN procedure can also be applied to Rocchio, thus obtaining Rocchio-ANN. However, the Rocchio and CoRocchio methods (without the additional ANN strategy) cannot be used in this circumstance. The main results are presented in Table 4.

The baseline methods exhibit the same trends observed on the original TREC DL query set (previous section), except that now the PRF methods for DRs are more consistent in the improvements they provide.

The results obtained when exploiting the historical click log with CoRoccho-ANN exhibits substantial improvements over the baselines. The trends observed for unseen queries are similar to those for logged queries (note that the absolute numbers cannot be compared across the two settings, because they refer to different query sets). The highest effectiveness is measured in the perfect and unbiased user setting has the highest effectiveness. When position bias is added to the perfect click behaviour, the effectiveness of DRs with Rocchio-ANN decreases. However, DRs with CoRocchio-ANN achieve the same effectiveness obtained when no position bias was present in the click signal. Similar results are found for noisy clicks, but click noise has more of a negative impact than position bias.

For the methods that exploits historic click log in Table 3, i.e., Rocchio and CoRocchio, they cannot directly deal with unseen queries because there is no associated click data in the log for them to update representation for unseen queries. However, our propose ANN approach for Rocchio and CoRocchio can deal with unseen queries by searching top-k nearest neighbor queries in the click log and exploits their clicked passages' representation. As the results reported in the Table 4, both Rocchio-ANN and CoRocchio-ANN shown large improvement over PRF baselines for all the user settings. Similar to the results for the logged query in Table 3, the perfect and unbiased user setting has the highest effectiveness for DRs uses both Rocchio-ANN and CoRocchio-ANN. When the position bias is added to perfect click behaviour, the effectiveness of DRs with Rocchio is deceased. However, DRs with CoRocchio-ANN achieve the same effectiveness as the setting without position bias. Similar results are found for noise click settings, where CoRocchio-ANN still can remove the position bias for the logged click data hence improving the effectiveness for DRs. Also similar to logged query results, click noise has more negative impact than position bias, and the worse effectiveness scores are observed with the user setting that both click noise and position bias is presented. Although, all the evaluation scores for unseen queries are lower than that of logged queries under the same experimental setting, the improvement of exploiting click feedback from the nearest neighbor queries in the click log is still impressive.

6 CONCLUSIONS AND OUTLOOK

In this paper we investigated how to exploit implicit feedback such as the click signal contained in historic click logs to improve the effectiveness of dense retrievers (DRs). With this respect, we proposed to adapt methods used for PRF with DRs to deal with implicit feedback rather than pseudo relevance feedback. To create and investigate effective methods for exploiting the click signal with DRs, we had to overcome three key challenges.

Challenge 1 related to the absence of a dataset that is sufficient for the training and evaluation of DRs and at the same time contains adequate click information to support our study. To address this challenge, we adapted evaluation practices from online and counterfactual LTR to datasets used for DR evaluation, such as TREC DL. This resulted in the ability of simulating user clicks on SERPs produced by DRs to collect a historic click log. This allowed us to investigate whether the historic click signal improves DRs effectiveness – we show that our Rocchio method does indeed effectively exploit the click signal and improves DRs effectiveness.

Related to Challenge 1, we also adapted a generative technique for query simulation to augment the click log with a larger set of queries with known relevance judgements. The availability of augmented queries allowed us to study the effectiveness of methods that exploit the historic click signal on queries that have not been observed beforehand.

Challenge 2 related to the presence of bias (and specifically position bias) in the click signal and its effect on the effectiveness of DR methods. With this respect we found that this bias does reduce the effectiveness of the proposed Rocchio method for exploiting the implicit feedback from historic clicks. To address this, we then devised the CoRocchio method, which counterfactually de-biases the click signal. Theoretical and empirical analyses demonstrated that CoRocchio can effectively address click bias.

Finally, Challenge 3 related to the fact that our counterfactual technique relies on having observed the current user query among those in the historic click log. This is a strong assumption: e.g., the majority of queries observed by web search engines are new queries. To deal with this challenge, we further refined our method to identify in the historic query log queries that are similar (but not identical) to the one currently at hand using the dense query representations. We found that CoRocchio-ANN is capable to effectively exploit the historic click signals of related queries to improve the dense representation (and thus effectiveness) of the current query, which was not observed in the click log.

Our study is the first that investigates how implicit click signal could be exploited in the context of DRs. We believe our work paves the way for the development of effective methods based on DRs that exploit click signals in an online manner. Code and experimental results are publicly available at https://github.com/ielab/Counterfactual-DR.

Acknowledgements. This research is partially funded by the Grain Research and Development Corporation project AgAsk (UOQ2003-009RTX).

REFERENCES

[1] Nasreen Abdul-Jaleel, James Allan, W Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. *Computer Science Department Faculty Publication Series* (2004), 189.

[2] Aman Agarwal, Soumya Basu, Tobias Schnabel, and Thorsten Joachims. 2017. Effective evaluation using logged bandit feedback from multiple loggers. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 687–696.

[3] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating position bias without intrusive interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 474–482.

[4] Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. 19–26.

[5] Negar Arabzadeh, Alexandra Vtyurina, Xinyi Yan, and Charles LA Clarke. 2021. Shallow pooling for sparse labels. *arXiv preprint arXiv:2109.00062* (2021).

[6] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click models for web search. *Synthesis lectures on information concepts, retrieval, and services* 7, 3 (2015), 1–115.

[7] Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. ORCAS: 20 million clicked query-document pairs for analyzing search. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2983–2989.

[8] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the TREC 2019 Deep Learning Track. In *Proceedings of the Twenty-Ninth Text REtrieval Conference (NIST Special Publication)*. National Institute of Standards and Technology (NIST).

[9] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the TREC 2019 Deep Learning Track. In *Proceedings of the Twenty-Ninth Text REtrieval Conference (NIST Special Publication)*. National Institute of Standards and Technology (NIST).

[10] Zhichong Fang, Aman Agarwal, and Thorsten Joachims. 2019. Intervention harvesting for context-dependent examination-bias estimation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 825–834.

[11] Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021. Complement lexical retrieval model with semantic residual embeddings. In *European Conference on Information Retrieval*. Springer, 146–160.

[12] Zhiwei Guan and Edward Cutrell. 2007. An eye tracking study of the effect of target rank on web search. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 417–420.

[13] Fan Guo, Chao Liu, and Yi Min Wang. 2009. Efficient multiple-click models in web search. In *Proceedings of the second acm international conference on web search and data mining*. 124–131.

[14] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. In *The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 113–122.

[15] Daniel G Horvitz and Donovan J Thompson. 1952. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association* 47, 260 (1952), 663–685.

[16] Rolf Jagerman, Harrie Oosterhuis, and Maarten de Rijke. 2019. To Model or to Intervene: A Comparison of Counterfactual and Online Learning to Rank from User Interactions. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. Association for Computing Machinery, 15–24.

[17] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. 133–142.

[18] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum*, Vol. 51. Acm New York, NY, USA, 4–11.

[19] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 781–789.

[20] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 6769–6781.

[21] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[22] Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 39–48.

[23] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 6086–6096.

[24] Hang Li, Ahmed Mourad, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. 2021. Pseudo Relevance Feedback with Deep Language Models and Dense Retrievers: Successes and Pitfalls. *arXiv preprint arXiv:2108.11044* (2021).

[25] Jimmy Lin and Xueguang Ma. 2021. A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques. *arXiv preprint arXiv:2106.14807* (2021).

[26] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: An easy-to-use Python toolkit to support replicable IR research with sparse and dense representations. *arXiv preprint arXiv:2102.10073* (2021).

[27] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2020. Distilling dense representations for ranking using tightly-coupled teachers. *arXiv preprint arXiv:2010.11386* (2020).

[28] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2021. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*. 163–173.

[29] Antonio Mallia, Omar Khattab, Torsten Suel, and Nicola Tonellotto. 2021. Learning passage impacts for inverted indexes. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1723–1727.

[30] Graham McDonald, Craig Macdonald, and Iadh Ounis. 2018. Active learning strategies for technology assisted sensitivity review. In *European Conference on Information Retrieval*. Springer, 439–453.

[31] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (CEUR Workshop Proceedings, Vol. 1773)*. CEUR-WS.org.

[32] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019).

[33] Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

[34] Douglas W Oard and William Webber. 2013. Information retrieval for e-discovery. *Information Retrieval* 7, 2-3 (2013), 99–237.

[35] Harrie Oosterhuis and Maarten de Rijke. 2017. Sensitive and scalable online evaluation with theoretical guarantees. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 77–86.

[36] Harrie Oosterhuis and Maarten de Rijke. 2018. Differentiable unbiased online learning to rank. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1293–1302.

[37] Harrie Oosterhuis and Maarten de de Rijke. 2021. Robust Generalization and Safe Query-Specializationin Counterfactual Learning to Rank. In *Proceedings of the Web Conference 2021*. 158–170.

[38] Zohreh Ovaisi, Ragib Ahsan, Yifan Zhang, Kathryn Vasilaky, and Elena Zheleva. 2020. Correcting for selection bias in learning-to-rank systems. In *Proceedings of The Web Conference 2020*. 1863–1873.

[39] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21 (2020), 1–67.

[40] Craig Silverstein, Hannes Marais, Monika Henzinger, and Michael Moricz. 1999. Analysis of a very large web search engine query log. In *Acm sigir forum*, Vol. 33. ACM New York, NY, USA, 6–12.

[41] Gabriela Surita, Rodrigo Nogueira, and Roberto Lotufo. 2020. Can questions summarize a corpus? Using question generation for characterizing COVID-19 research. *arXiv preprint arXiv:2009.09290* (2020).

[42] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*.

[43] HongChien Yu, Chenyan Xiong, and Jamie Callan. 2021. Improving Query Representations for Dense Retrieval with Pseudo Relevance Feedback. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*. ACM.

[44] Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 1201–1208.

[45] Yisong Yue, Rajan Patel, and Hein Roehrig. 2010. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *Proceedings of the 19th international conference on World wide web*. 1011–1018.

[46] Haotian Zhang, Gordon V Cormack, Maura R Grossman, and Mark D Smucker. 2020. Evaluating sentence-level relevance feedback for high-recall information retrieval. *Information Retrieval Journal* 23, 1 (2020), 1–26.
[47] Shengyao Zhuang, Zhihao Qiao, and Guido Zuccon. 2022. Reinforcement Online Learning to Rank with Unbiased Reward Shaping. *arXiv preprint arXiv:2201.01534* (2022).
[48] Shengyao Zhuang and Guido Zuccon. 2020. Counterfactual Online Learning to Rank. In *European Conference on Information Retrieval*. Springer, 415–430.
[49] Shengyao Zhuang and Guido Zuccon. 2021. Fast passage re-ranking with contextualized exact term matching and efficient passage expansion. *arXiv preprint arXiv:2108.08513* (2021).
[50] Masrour Zoghi, Tomáš Tunys, Lihong Li, Damien Jose, Junyan Chen, Chun Ming Chin, and Maarten de Rijke. 2016. Click-based hot fixes for underperforming torso queries. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 195–204.