Algorithmic Vibe in Information Retrieval

Ali Montazeralghaem
University of Massachusetts Amherst
Amherst, MA, USA
montazer@cs.umass.edu

Nick Craswell
Microsoft
nickcr@microsoft.com

Ryen W. White
Microsoft
ryenw@microsoft.com

Ahmed H. Awadallah
Microsoft
hassanam@microsoft.com

Byungki Byun
Microsoft
bybyun@microsoft.com

ABSTRACT

When information retrieval systems return a ranked list of results in response to a query, they may be choosing from a large set of candidate results that are equally useful and relevant. This means we might be able to identify a difference between rankers A and B, where ranker A systematically prefers a certain type of relevant results. Ranker A may have this systematic difference (different "vibe") without having systematically better or worse results according to standard information retrieval metrics. We first show that a vibe difference can exist, comparing two publicly available rankers, where the one that is trained on health-related queries will systematically prefer health-related results, even for non-health queries. We define a vibe metric that lets us see the words that a ranker prefers. We investigate the vibe of search engine clicks vs. human labels. We perform an initial study into correcting for vibe differences to make ranker A more like ranker B via changes in negative sampling during training.

KEYWORDS

Fairness, Gender Bias, Position Bias

ACM Reference Format:
Ali Montazeralghaem, Nick Craswell, Ryen W. White, Ahmed H. Awadallah, and Byungki Byun. 2023. Algorithmic Vibe in Information Retrieval. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3543507.3583384

1 INTRODUCTION

Bias in machine learning has been studied in application areas such as banking, healthcare, criminal justice, hiring, facial recognition, and others [29]. Unfair outcomes that arise in such cases might be due to bias in the data. For example, if there were already a bias against women in a historic dataset of human hiring decisions, a machine learned hiring predictor might also be biased against women. Machine learning algorithms may also create their own unfair outcomes or amplify the existing bias in the training data. When humans are in the loop using an ML system, they can reinforce existing biases and introduce their own. For example, in a Web search engine, users tend to click the results that were surfaced by the existing ML algorithm [11, 14, 20], and among those top-ranked results, they will also prefer results that are most "attractive" [47]. These clicks can then become training data for the next iteration of the ML model.

Responsible AI standards tend to suggest checking for bias, explaining model predictions, and monitoring model outcomes with human review.[1] Understanding, detecting, and monitoring bias allow the machine learning practitioner to take action, attempting to correct the bias. They can particularly focus on cases where the bias could cause harm.

For information retrieval systems such as Web search engines, we are not yet at the stage where we have standard tools for detecting and monitoring bias. It is also possible for us still to be surprised by new sources of bias, which may be application-dependent. We may discover different potential biases in an e-commerce website search, a Web-scale engine, and a health-related website search. Different services have different purposes and different user audiences around the world. Each user audience may have preferences for particular topic areas, content publishers, or clickbait terms. A machine-learned ranker, trained from such clicks, will return more results with those topics, publishers, and terms. This general pattern that the ML model is powerful enough to learn a bias is true whether the training data is user clicks or labels from relevance judges.

Studies have already started to identify biases that are detectable in specific search systems. It's possible for a search system to have gender bias when trained on label data that prefers male results, and applying a more powerful machine learning model intensifies the bias [40]. When users are searching for information and already have some belief of what the answer is, they are more likely to click a result that confirms their bias [45]. The problem is that there are many potential types of bias in different search systems on the Web and elsewhere, and the kinds of bias that arise may be application-dependent. This suggests that human review is needed to identify potential biases and harms, and rather than looking at a particular bias only (such as gender bias), the reviewer should be seeing a general overview of the patterns in search results, whatever they may be.

WWW '23, April 30–May 04, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583384

[1] Links to Responsible AI standards: • https://blogs.microsoft.com/wp-content/uploads/prod/sites/5/2022/06/Microsoft-Responsible-AI-Standard-v2-General-Requirements-3.pdf • https://ai.google/responsibilities/responsible-ai-practices/ • https://aws.amazon.com/machine-learning/responsible-machine-learning/

This paper defines the *vibe* of a search system as a relative comparison to a baseline search system. For example, comparing a more powerful model, which may be amplifying bias, to a less powerful baseline. For example, comparing a model trained on clicks to a model not trained on clicks. The vibe is measured based on first running a large number of queries in the two systems. Then we suggest several prototype vibe metrics, which compare the paired results of the systems and look for consistent differences that are query-independent. The goal is to find the largest and most systematic differences across the query set for human review.

We present three case studies, one using publicly available deep learning rankers, showing we can compare rankers that are readily available regardless of their architecture by comparing the vocabulary differences in their paired output rankings. In the second case study, we show that adding a health/medical-related vibe to a ranker is possible. The other case study compares proprietary rankers in Web search that have the same architecture but were trained on different types of data. In these case studies, we show that differences in the underlying training data are reflected in differences in ranker vibe when we run a separate set of test queries. We quantify the user impact in simple ways; for example, given the vibe difference, how often do we also observe a difference in the top-ranked results that users see? Finally, having detected a vibe difference, in our open source case study we demonstrate that we can train the ranker with different negative sampling, to reduce the vibe difference from the baseline ranker, as a proof of concept that vibe is not only detectable but correctable.

2 RELATED WORK

2.1 Gender Bias in Information Retrieval

Rekabsaz et al. [40] recently introduced a framework for measuring gender bias in retrieval models. They defined a set of highly gender-related words to measure document female/male, namely the degree of female/male-related concepts in a document. Also, they represented two retrieval gender bias metrics. They provided a set of non-gendered queries among the queries of the development set of MS MARCO passage retrieval collection. They showed that the neural ranking models have higher degrees of bias than the BM25 retrieval model. Their results show the existence of significant gender bias (towards males) in the retrieved documents of all the models and confirm that the neural ranking models, despite better retrieval performance, overall intensify gender bias in retrieval results toward males compared with BM25.

Bigdeli et al. [7] proposed a simple yet effective sampling strategy for training ranking models that would allow the rankers to maintain their retrieval effectiveness while reducing gender biases. By using this intuition that neural ranking models are quite sensitive to the adopted negative sampling strategy, the authors proposed a systematic negative sampling strategy, exposing the neural rankers to representations of gender bias that need to be avoided when retrieving documents.

Fabris et al. [15] proposed the Gender Stereotype Reinforcement (GSR) metric to quantify the tendency of a search engine ranked list support gender stereotypes. GRS exploits gender bias encoded in Word Embeddings [30]. The authors examine information retrieval ranking algorithms from different families and measure each system's performance and GRS on TREC collections [22].

Zhao et al. [51] analyzed gender bias in ELMo's contextualized word vectors [41]. The authors conducted that training data for ELMo contains significantly more male than female entities, and trained ELMo embeddings encode gender information. They showed that ELMo encodes information about male and female entities' genders differently. Otterbacher et al. [37] examined how personality characteristics affect the ability of the users to recognize gender-biased image search results. Chen et al. [9] investigated gender inequality in resume search engines. The authors showed negative correlations between rank and inferred gender in their dataset. They concluded that even when controlling for all other visible candidate features, there is a slight penalty against feminine candidates.

2.2 Position Bias in Information Retrieval

Multiple eye-tracking and other empirical investigations have shown position bias in search rankings [11, 14, 20]. These studies show that top-ranked results are much more likely to be viewed and clicked than those at lower ranks. The effect still exists even if the items in various rankings are randomly permuted [25]. These findings prompted the development of numerous click models and other position bias-aware re-ranking techniques [10, 26, 43]. Wang et al. [44] studied the problem of position bias estimation for unbiased learning-to-rank algorithms. They proposed a new regression-based expectation-maximization (EM) algorithm to estimate position bias. Ai et al. [3] proposed a dual learning algorithm for automatic unbiased learning to rank. The proposed algorithm jointly learns unbiased propensity and ranking models from user clicks. Wang et al. [43] introduced a new unbiased learning-to-rank framework by addressing the bias as a counterfactual effect. Agarwal et al. [2] presented the first method for generating reliable propensity estimates without manual relevance assessments, disruptive interventions, or restrictive relevance modeling assumptions.

2.3 Fairness

Concerns regarding fairness and bias have lately increased due to the increasing prevalence of data-driven learning models in algorithmic decision-making[2].

Dixon et al. [13] introduced methods to quantify and mitigate unintended bias in text classification models. The authors defined a working definition of unintended bias in the classification task. Based on their definition, a model contains unintentional bias if it performs better for comments about some groups than others. They proposed a simple and novel technique to reduce the bias by strategically adding data. They showed this strategy could mitigate the unintended biases in a model without harming the overall model quality.

Grag et al. [19] provided a counterfactual token fairness measure for counterfactual computing fairness in the text classification task. The authors proposed three methods for counterfactual token fairness as follows: 1) blindness which substitutes all identity tokens with a special IDENTITY token; 2) counterfactual augmentation, which involves augmenting the model's training set with generated counterfactual examples; and 3) counterfactual logit pairing

[2]https://facctconference.org/

which encourages the model to be robust to identity by adding a robustness term to the training loss.

Zafar et al. [48] proposed a new notion of unfairness, and disparate mistreatment, which is defined in terms of misclassification rates. The authors then suggested practical metrics for decision boundary-based classifiers that account for disparate maltreatment. Zemel et al. [50] proposed a learning algorithm for fair classification that attains both group and individual fairness. Abede et al. [1] investigated mechanisms for fair division of resources via social comparison.

2.3.1 Fairness in Ranking. Fair-ranking research from the past is rare and recent. Yang et al. [46] showed how to incorporate various notions of group fairness into ranking quality measures. Zehlike et al. [49] introduced methods that maintain a high level of ranking quality while diversifying the ranking results in terms of the inclusion of members from various groups in the ranking prefixes. Celis et al. [8] studied this problem from a theoretical perspective, with the results provided for the problem's computational complexity. Singh and Joachims [42] introduced a notion of group fairness based on equality of exposure for demographic groups.

Rekabsaz et al. [39] provided a novel framework to measure the fairness in the retrieved text contents of ranking models. They also proposed a novel fairness metric of a ranked list. Then, they introduced a new model to mitigate biases by approaching deep retrieval models with an adversarial training method. The authors showed that the fairness of BERT rankers significantly improves by applying the proposed adversarial training.

3 METHODOLOGY

When ranker A and ranker B return relevant results, they can return different "flavor" or "feel". For example, for a health query, there is a big difference between a relevant document from the CDC (Centers for Disease Control and Prevention) and a relevant document that is from a questionable source and trying to sell you a miracle cure. Both are topically relevant, but a ranker can be trained that systematically favors one type of relevance or another.

In this section, we first define the systematic difference between two rankers, called "vibe". Then we provide different approaches and tools to characterize, quantify and even correct the vibe.

3.1 Vibe: Systematic Difference Between two Rankers

In an information retrieval system, for example, a search engine, we are supposed to rank a collection of items and show the most relevant of them to the user. The number of top items we can show the user in each step varies based on applications. For example, in a search engine, we typically provide the top 10 documents on the first page to the user, and these top 10 documents can significantly impact user satisfaction with the system.

By running a given query in different information retrieval systems (i.e., rankers), we might get different rank lists (e.g., top 10 documents) but equally relevant to the user's query. If this difference is systematic and not random, we might be able to detect it and even correct it if needed. In other words, two rankers might systematically promote different types of relevant documents across many queries. We call these kinds of differences between two rankers

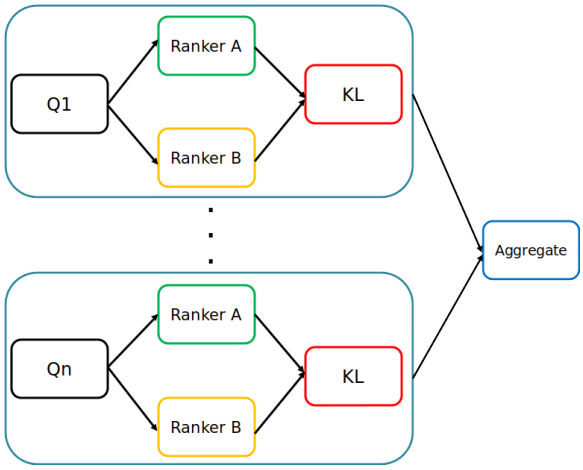

Figure 1: An illustration of approach one.

"vibe"[3]. For example, a ranker tends to show more health-related relevant documents to the user. Gender bias [40] in retrieval models can be considered as a specific example of the vibe. Another example of detecting vibe is finding the difference between two rankers, where rankers A and B are trained on clicks vs. judgments, respectively.

3.2 Detecting Vibe

In this section, we propose three approaches for detecting vibe differences between two rankers.

Let A and B be two publicly available rankers. To detect the vibe difference between two rankers, we first need to specify a large set of queries $Q = \{q_1, q_2, ..., q_{|Q|}\}$. For each query, we get the top N documents retrieved by each ranker e.g., $D_{(q_i,A)} = \{d_1, d_2, ..., d_N\}$. Using these ranking lists for all queries computed by two rankers, we propose three approaches to calculate the vibe difference between two rankers in the following sections. The first two approaches use the Kullback-Leibler (KL) divergence between two relevance-based language models to find the contribution of each term in a ranker compared to another. In other words, these two approaches use the KL divergence method to compare the relevance-based language model of different rankers, which is calculated differently in the two approaches. The third approach is a novel method based on term frequency competition to find the most contributed terms in finding the vibe difference between two rankers.

3.2.1 ***Approach one: KL-Agg.*** In this approach, we first compute the relevance-based language model [5, 23, 27, 32–35] for a ranking list returned by each ranker for each query, i.e., $D_{(q_i,A)}$ as follows:

$$\theta_{(q_i,A)} = RM1(D_{q_i,A}) \quad (1)$$

where $RM1(.)$ is a well-known and state-of-the-art variant of the relevance-based language models proposed by Lavrenko and Croft [27]. Our goal is to compare language models of two rankers to see how ranker A differs from ranker B across many queries. To achieve this goal, we use "KL contribution" to compare two relevance-based

[3]Vibe definition at Merriam-Webster Unabridged Dictionary: "A distinctive feeling or quality capable of being sensed".

language models, i.e., $\theta_{(q_i,A)}$ and $\theta_{(q_i,B)}$, and find out the terms contribute most to the KL divergence of B from A (or vice versa). Given relevance-based language models of two rankers for a query, their KL divergence is:

$$KLD(\theta_{(q_i,A)}, \theta_{(q_i,B)}) = \sum_{T \in \mathcal{T}} P(T|\theta_{(q_i,A)}).\log(\frac{P(T|\theta_{(q_i,A)})}{P(T|\theta_{(q_i,B)})}), \quad (2)$$

where $\mathcal{T}$ is the set of all terms in the relevance-based language model of the ranker A for query q_i, and $P(.)$ denotes term T's probability of occurrence under the distribution $\theta_{(q_i,A)}$ and $\theta_{(q_i,B)}$.

To find the impact of a term T in detecting vibe differences between a ranker A and a ranker B, we can aggregate Eq.2 over the query set Q as follows:

$$KLD(T_{AB}) = \sum_{q \in Q} P(T|\theta_{(q,A)}).\log(\frac{P(T|\theta_{(q,A)})}{P(T|\theta_{(q,B)})}). \quad (3)$$

$KLD(T_{AB})$ computes the impact of each term T in the ranker A compared to the ranker B over all queries. In our experiments, we find that there is a chance that a term has a high impact on both rankers compared to another one, i.e., has a high value in two directions $KLD(T_{AB})$ and $KLD(T_{BA})$. The reason is that a word may be a high-impact term in some queries in one ranker and has a high score in other queries in another ranker. To solve this problem, we compute Eq.3 in both directions (i.e., ranker A compared to ranker B and vice versa) and get a difference to compute the true impact of a term in a ranker compared to another one as follows:

$$\begin{aligned}
Imp(T_{AB}) &= KLD(T_{AB}) - KLD(T_{BA}) \\
&= \sum_{q \in Q} P(T|\theta_{(q,A)}).\log(\frac{P(T|\theta_{(q,A)})}{P(T|\theta_{(q,B)})}) \\
&- \sum_{q \in Q} P(T|\theta_{(q,B)}).\log(\frac{P(T|\theta_{(q,B)})}{P(T|\theta_{(q,A)})}) \\
&= \sum_{q \in Q} P(T|\theta_{(q,A)}).\log(\frac{P(T|\theta_{(q,A)})}{P(T|\theta_{(q,B)})}) \\
&+ \sum_{q \in Q} P(T|\theta_{(q,B)}).\log(\frac{P(T|\theta_{(q,A)})}{P(T|\theta_{(q,B)})}) \\
&= \sum_{q \in Q} (P(T|\theta_{(q,A)}) + P(T|\theta_{(q,B)})).\log(\frac{P(T|\theta_{(q,A)})}{P(T|\theta_{(q,B)})}).
\end{aligned} \quad (4)$$

$Imp(T_{AB})$ has a high value for terms that have high KL contribution in the ranker A but have a low impact in another direction (i.e., in the ranker B compared to the ranker A).

Using the impact function for terms, we compute a vibe metric to find the vibe difference between rankers A and B as follows:

$$Vibe(A, B) = \frac{1}{|Q|} \sum_{T \in \mathcal{T}_A} Imp(T_{AB}), \quad (5)$$

where $\mathcal{T}_A$ is the set of all terms in the top retrieved documents for all queries by ranker A.

Likewise, we can define $Vibe(B, A)$ as follows:

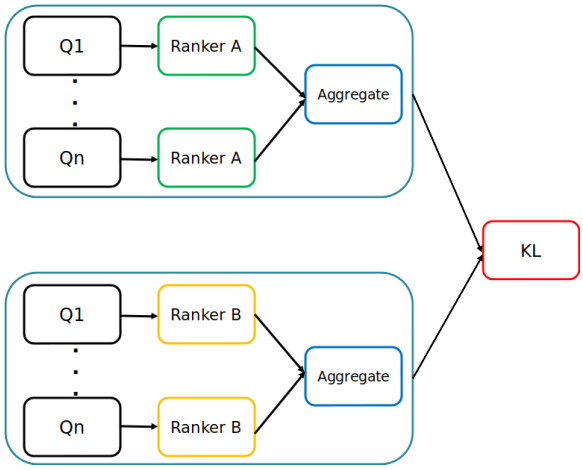

Figure 2: An illustration of approach two.

$$Vibe(B, A) = \frac{1}{|Q|} \sum_{T \in \mathcal{T}_B} Imp(T_{BA}). \quad (6)$$

Figure 1 illustrates the approach one workflow.

3.2.2 ***Approach two: Agg-KL.*** In this approach, we first aggregate relevance-based language models of all queries to build a unified language model for each ranker as follows:

$$\theta_{(Q,A)} = \frac{\theta_{(q_1,A)} + \theta_{(q_2,A)} + ... + \theta_{(q_{|Q|},A)}}{|Q|}, \quad (7)$$

and then compute the KL divergence between them to find the vibe difference between two rankers.

By computing $\theta_{(Q,A)}$, we can show the impact of a term in a ranker compared to another directly. Therefore, the vibe difference between two rankers can be computed as follows:

$$Vibe(A, B) = \sum_{T \in \mathcal{T}_A} KLD(\theta_{(Q,A)}, \theta_{(Q,B)}). \quad (8)$$

Figure 2 shows the approach two workflow.

3.2.3 ***Approach three: TF Competition.*** Approach three is different than the previous two approaches. This approach treats each query as a competition between two rankers, where a high term frequency (TF) in the retrieved documents wins. More specifically, to compute the impact of a term, we count the number of queries that the TF of that term was higher in a ranker than another. For example, for the word "gov", there were 50 queries where TF was higher than in a ranker compared to another, and 200 queries where the TF was higher than another. In our experiments, we find that this approach can effectively find terms that are promoted by a ranker across many queries.

3.3 Vibe Examples

This section shows some examples of vibe differences between the two rankers. We use the development (dev) set consisting of 6,980 *general* queries from the MS MARCO[4] [6] passage dataset to get the results of two first examples, i.e., examples 1 and 2. For

[4]MicroSoft MAchine Reading COmprehension

the third example, we use a set of 6,082 English queries from a major web search engine to show the vibe difference between click-based and judgment-based proprietary rankers. We show these examples using our first approach, i.e., KL-Agg, but we had similar observations using two other approaches.

3.3.1 *Example 1: Comparing two publicly available rankers on non-health/medical queries.*

For detecting the vibe difference between two rankers, we select two publicly available neural rankers: 1) Vanilla BERT (VBERT) [12], which is a contextualized language model and a powerful neural model that has been shown to be effective for ranking, and 2) MonoT5 [36], which scores documents using a causal language model. The first model, i.e., VBERT, is trained on health/medical-related queries from the MS MARCO dataset, and MonoT5 is trained on general queries. Our goal in this example is to show that there is a health-related vibe in VBERT compared to MonoT5 when we need to run non-health queries to these systems. In other words, VBERT promotes health-related documents even if we run a non-health query.

We use VBERT and MonoT5 to run the dev set consisting of 6,980 *general* queries from the MS MARCO passage dataset [6]. The most contributed terms and their scores computed by the first approach Eq.4 are reported in Table 1. According to this table, our approach can detect health/medical words (e.g., "medic", "organ", or "bodi", note that the words are stemmed using the Porter stemmer) when we compare VBERT against MonoT5. This shows that the VBERT model has a vibe related to health and prefers to select medical or health-related documents. This can be very harmful to the system, especially when we need to run a non-health query to the retrieval system with a vibe related to health/medical. As an example, for a non-health query "what is an aml surveillance analyst", the ten most contributed terms in the vibe difference between VBERT and MonoT5 are {'blood', 'bloodforming', 'cell', 'initi', 'myeloid', 'develop', 'type', 'leukemia', 'lymphocyt', 'procedur'}, which are mostly about health/medical.

If we compare MonoT5 against VBERT, the most contributed words are often numbers or words related to numbers (e.g., "5", "10", or "vote"). This shows that the MonoT5 has a vibe related to numbers for retrieving documents.

3.3.2 *Example 2: Adding a vibe to a ranker.*

In this example, we aim to add a health/medical-related vibe to the MonoT5 model. More specifically, by adding 2000 medical-related queries (which are randomly selected from 78895 medical-related queries from the MS MARCO dataset) to the training of MonoT5 (named MediMonoT5), this model prefers to retrieve documents that have medical topics, even for non-health/medical queries.

Our results for this example are reported in Table 2. Like the previous example, we report the most contributed terms and their scores in this Table. As a first observation, the impact scores of terms, even for most contributed terms, are very small, e.g., $8e-5$. The reason is that the difference between the two models is very small to ensure that the performance of the two systems is the same. In other words, we get the exact same performance for two models by running the dev set from the MS MARCO dataset but with a different vibe. According to this table, our approach detects medical words by comparing MediMonoT5 against MonoT5, e.g., "eunuch", "incid", and "symptom". This means that MediMonoT5 prefers to select documents with medical words more than MonoT5 under the same conditions. This shows that we can add an arbitrary vibe to a model by adding a topic-specific query set to the model's training. This can be useful for retrieving documents for queries in a topic but harmful for retrieving documents for general queries.

Table 1: Example 1, most contributed terms and their scores in KL divergence between VBERT and MonoT5 based on Eq.4. Words are stemmed using the Porter stemmer.

VBERT VS. MonoT5	MonoT5 VS. VBERT
('definit',19.7), ('dictionari',10.4),	('answer',8.0), ('noun', 7.2),
('type',8.4), ('exampl',7.0),	('sai',6.3), **('5',6.0)**,
('refer',6.4), ('depend',5.5),	**('10',5.8)**, **('vote',4.7)**,
('medic',5.3), **('organ',5.1)**,	**('0',4.7)**, ('now',4.6),
('translat',4.7), **('bodi',4.7)**,	('don',4.3), **('000',3.8)**,
('function',4.3), ('typic',4.2),	('see',3.5), ('think',3.4),
('term',4.2), **('temperatur',4.1)**,	('home',3.3), ('just',3.2),
('food',4.1), ('occur',4.0),	**('percent',3.2)**, **('12',3.1)**,
('made',3.9), **('diet',3.7)**,	('onli',3.1), **('9',2.9)**,
('unit',3.7), ('salari',3.6),	('still',2.8), **('degree',2.7)**,
('english',3.5), ('high',3.5),	**('15',2.6)**, **('22',2.6)**,
('largest',3.5), **('symptom',3.5)**	('happen',2.5), ('ve',2.4)

Table 2: Example 2, adding some medical-related queries to the training of MonoT5 to change its vibe without decreasing its performance. Words are stemmed using the Porter stemmer.

MediMonoT5 VS. MonoT5	MonoT5 VS. MediMonoT5
('eunuch',8e-5), ('mai',6e-5),	('noun',7e-5), ('state',6e-5),
('record',6e-5), ('new',6e-5),	('consid',6e-5), ('stand',6e-5),
('earli',6e-5), **('incid',6e-5)**,	('brought',6e-5), ('lab',6e-5),
('work',5e-5), ('world',5e-5),	('dai',5e-5), ('live',5e-5),
('process',5e-5), **('food',5e-5)**,	('problem',5e-5), ('anim',5e-5),
('chang',5e-5), **('symptom',5e-5)**,	('john',5e-5), ('count',5e-5),
('grow',5e-5), ('articl',5e-5),	('rise',5e-5), ('anti',5e-5),
('devic',5e-5), ('relat',5e-5),	('onli',4e-5), ('found',4e-5),
('true',5e-5), **('vessel',5e-5)**,	('month',4e-5), ('requir',4e-5),
('dzuma',5e-5), **('ascot',5e-5)**,	('group',4e-5), ('make',4e-5),
('amina',5e-5), ('5',4e-5),	('side',4e-5), ('now',4e-5),
('averag',4e-5), ('made',4e-5)	('pai',4e-5), ('full',4e-5)

3.3.3 *Example 3: Comparing click-based and judgment-based rankers.*

In this case study, we compare two proprietary rankers in Web search that have the same architecture but were trained on two different types of data, i.e., clicks and judgments. For this case study, we use 6,082 English queries from a major web search engine.

The results of our first approach for this case study are reported in Table 3. The interesting observations in these results are as follows:

- **Sites that require the login penalized by judges**: An interesting observation here is seeing words like: "Quizlet", "newgen", "tsp", "myhsc", and "irctc". For example, Quizlet is a website used for studying and learning, and one must

Table 3: Example 3, most contributed terms and their scores in KL divergence between click-based and judgment-based rankers.

Click VS. Judgment	Judgment VS. Click
(**'quizlet',4e-5**),	('jhini',3e-5),
(**'newgen',3e-5**),	('bsel',3e-5),
('5c',2e-5),	('quote',3e-5),
('hov',2e-5),	('latinboysextoy', 2e-5),
(**'flashcards',2e-5**),	(**'chebi',2e-5**),
('profuse',2e-5),	('barchart',2e-5),
('leave',2e-5),	('totowa',2e-5),
('limited',2e-5),	(typescript, 2e-5),
('perino',2e-5),	(**'dictionary',2e-5**),
(**'tsp',2e-5**),	('góra',2e-5),
('d5',2e-5),	('mehboob',2e-5),
('inprivate',2e-5),	('hatim',2e-5),
('l08',2e-5),	('zacks',2e-5),
(**'fanpage20',2e-5**),	('silsoft',2e-5),
('angler',1e-5),	('cooktop',2e-5),
(**'delta',1e-5**),	('irrigated',1e-5),
(**'quiz',1e-5**),	('hollins',1e-5),
('befikre',1e-5),	('multiline',1e-5),
('kasei',1e-5),	('induction',1e-5),
(**'col3negtelevision',1e-5**),	('iso100',1e-5),
('tl',1e-5),	('f44',1e-5),
(**'myhsc',1e-5**),	('fitchburg',1e-5),
('volume',1e-5),	('heterogenitet', 1e-5),
(**'irctc',1e-5**)	('rbc',1e-5)

sign up or log in to use it. So, users can find this website useful and click on it; however, human judgment labels it as a non-relevant website.

- **Dictionary-style sites overrated by judges**: As another observation, we find that the dictionary-style sites such as "chebi"[5] are preferred by judges. In other words, the judges overrate dictionary-style sites, and the reason is that judges label a site that has provided a definition for the user's query as relevant content.
- **Less authoritative sites are preferred in clicks:** Less authoritative sites like blogs, forums, and entertainment such as "col3negtelevision" and "fanpage20" are more present in the click-based system. These are results that users choose to click on, but the relevance judges are trained to be skeptical of some forms of user-generated content.

4 CORRECTING VIBE

After detecting the vibe, in this section, we aim to correct the ranker's vibe to get the same or even better performance. To achieve this goal, we need to use an approach that does not require any change to the architecture of a ranker. We propose a straightforward but efficient sampling method for training a ranker that would permit the ranker to keep its retrieval efficiency while decreasing the vibe. More specifically, we provide a systematic negative sampling strategy that exposes the ranker to vibe representations that should be avoided while retrieving documents. Bigdeli et al. [7] introduced a similar approach to reduce gender biases in neural rankers.

To train a neural ranker, we provide two sets of positive and negative samples for each query to teach the ranker to prefer positive samples over negative ones. Most neural rankers utilize either the top-retrieved documents by a fast ranker such as BM25 [4, 16, 31] or random documents as a set of negative documents [17, 18, 21, 28, 38].

Let $\mathcal{D}_{Rand}$ be the original pool for selecting negative samples in training a neural ranker. We define another source set for selecting negative samples $\mathcal{D}_{Vibe}$, which includes all negative documents with high-impact terms in the vibe of a ranker. For example, if we already know that a ranker has a health vibe and promotes documents with the "organ" word, $\mathcal{D}_{Vibe}$ would be a set of documents containing the "organ" term. To choose these high-impact terms to use in selecting negative sampling documents, we get the vibe difference between the ranker A and a baseline search system such as BM25 using Eq.5 and find the top n terms as follows:

$$\mathcal{W}_n = \{T_i \in sorted(\mathcal{T}_A) | \quad 1 \leq i \leq n\} \tag{9}$$

where $sorted(.)$ returns a sorted list in descending order based on impacts of words computed by Eq.4. Then, we utilize these words in selecting negative documents as follows:

$$\mathcal{D}_{Vibe} = \{d \in \mathcal{D}_{Rand} | \quad \mathcal{W}_n \cap d \neq \emptyset\}. \tag{10}$$

According to this question, we select negative documents that contain at least one word from high-impact terms in the vibe difference between the ranker A and BM25, i.e., $\mathcal{W}_n$.

To form the final negative samples, we use a free-parameter λ, which shows what percentage of the documents are selected from the vibe documents $\mathcal{D}_{Vibe}$ and the rest (i.e., $1-\lambda$) from the original pool $\mathcal{D}_{Rand}$.

5 EXPERIMENTS

5.1 Experiments for correcting vibe

For these experiments, we aim to that train two models with identical positive samples but different negative samples. To achieve this goal, we trained two models based on Vanilla BERT, i.e., VBERT (which is pre-trained for the task of language modeling and next sentence prediction) [12] as follows:

- Model one: **Tuned-VBERT**
  - Use VBERT and fine-tune it using MS MARCO medical queries (78, 895 queries) with random negative sampling.
  - Evaluate the trained model on general queries on MS MARCO dev set consisting of 6, 980 queries.
- Model two: **Tuned-VBERT-Neg**
  - Use VBERT and fine-tune it using MS MARCO medical queries with *not completely random negative samples*, i.e., use $\mathcal{D}_{Vibe}$ set to include negative documents based on their vibe.
  - Evaluate the trained model on general queries on the MA MARCO dev set consisting of 6, 980 queries.

[5]Chemical Entities of Biological Interest (ChEBI) is a freely available dictionary of molecular entities focused on 'small' chemical compounds

Table 4: Comparison of the proposed negative sampling and baselines. The superscript ▲ indicates that the improvements BM25 and Tuned-VBERT statistically significant. λ is equal to 0.8 in all experiments. It means that 20 percent of negative samples are selected randomly.

Model Type	Model Name	n	MRR	MAP	P@10	NDCG@10	Vibe(Model,BM25)
Word Based Retrieval	BM25	-	0.1953	0.1920	0.0402	0.2299	-
Neural Ranking Model by Random Negative Sampling	Tuned-VBERT	-	0.2558	0.2516	0.0495	0.2985	**0.2204**
Our Approach	Tuned-VBERT-Neg	1	0.2648▲	0.2608▲	0.0509	0.3090▲	0.1416
	Tuned-VBERT-Neg	5	0.2637▲	0.2591	0.0512	0.3087▲	0.1442
	Tuned-VBERT-Neg	15	0.2582	0.2546	0.0506	0.3036	0.1339
	Tuned-VBERT-Neg	25	**0.2711**▲	**0.2671**▲	0.0506	**0.3133**▲	0.1477
	Tuned-VBERT-Neg	35	0.2645▲	0.2601▲	**0.0513**	0.3094▲	0.1514

Comparing these two models could be a clean way to show the vibe can be controlled. We implemented and trained our model using PyTerrier[6] and OpenNIR[7].

As the baseline for selecting the top n high-impact terms in Eq.9, we use BM25, a fast and unsupervised ranking model. Therefore, we compute the vibe difference between the model trained on medical queries, i.e., Tuned-VBERT and BM25 using Eq.5, and find n high-impact terms by Eq.4 and use them in negative sampling. For example, if $n = 5$, then $\mathcal{W}_n = \{'unit', 'state', 'popul', 'includ', 'censu'\}$ (Note the terms are stemmed by the Porter stemmer) and we select documents that have at least one of these terms as negative samples. In our experiments, we find that the best value of λ (the percentage of documents selected from the vibe documents $\mathcal{D}_{Vibe}$) is 0.8. In other words, we need to select 20 percent of negative samples randomly.

5.1.1 ***Evaluation Measures.*** For evaluating the performance of the models, we use mean average precision (MAP) of the top 1000 documents, mean reciprocal rank (MRR), normalized discounted cumulative gain (NDCG) at 10 [24], and precision of the top 10 retrieved documents (P@10). Statistically significant differences of performance are determined using two-tailed paired t-test at 95% confidence level ($p\_value < 0.05$). We also report the vibe metric between each model and BM25.

5.1.2 ***Results and Discussion.*** The results of this experiment are reported in Table 4. This table reports Tuned-VBERT results with different values for n. The negative sampling process is the only difference between Tuned-VBERT and Tuned-VBERT-Neg models, and all other parameters are the same in both models. We aim to reduce the vibe in the Tuned-VBERT while not reducing the model's performance. The ideal case would be reducing the vibe in a model and, at the same time, increasing its' performance. According to Table 4, the Tuned-VBERT-Neg model with different values for n outperforms Tuned-VBERT in terms of retrieval measures, i.e., MAP, MRR, P@10, and NDCG@10.

An interesting observation is that if we can find just one high-impact term $n = 1$ and use it to select negative samples by Eq.10, this would decrease the vibe of the model and, at the same time, increase the system's performance for general queries. This shows that the most contributed term in the vibe difference between Tuned-VBERT and BM25 helps to find the most helpful negative samples and use them in the model's training.

[6]https://github.com/terrier-org/pyterrier
[7]https://github.com/Georgetown-IR-Lab/OpenNIR

Table 5: Toning down the vibe of Tuned-VBERT (VBERT in short) by comparing this model with BM25 and using 15 high-impact terms: {"unit", "state", "popul", "includ", "censu", "answer", "time", "citi", "counti", "2010", "town", "area", "languag", "cell", "seat"} to select negative samples for training Tuned-VBERT-Neg (VBERT-Neg in short). Words are stemmed using the Porter stemmer.

VBERT VS. VBERT-Neg	VBERT-Neg VS. VBERT
(**'answer',15.7**), (**'time',11.9**),	('definit',17.2), ('noun',12.7),
('first',10.7), ('best',7.9),	('bodi',12.5), ('type',11.0),
(**'popul',7.8**), ('question',7.3),	(**'cell',7.7**), ('creat',6.6),
(**'citi',7.1**), ('find',7.1),	('dictionari',6.5), ('organ',6.4),
('dai',6.1), ('averag',6.0),	('4',5.9), ('acronym',5.8),
(**'counti',6.0**), ('exampl',5.9),	('medic',5.7), ('form',5.7),
('surround',5.6), (**'includ',5.5**),	('caus',5.7), ('peopl',5.6),
('know',5.1), ('sai',5.0),	('group',5.6), ('system',5.5),
(**'town',4.9**), ('origin',4.8),	('design',5.3), ('stand',5.2),
('see',4.7), ('recommend',4.7),	('program',5.1), ('mean',5.1),
('take',4.5), ('common',4.5),	('muscl',4.6), ('made',4.5),
('well',4.5), ('record',4.5)	('contain',4.4), ('call',4.4)

By increasing the number of high-impact terms, i.e., n, we see improvements in the performance, but we don't see many changes in the vibe differences. The reason is that a high-impact word can strongly correlate with other vibe words (words that have high impacts on the vibe difference), and by choosing a few terms, we are considering other vibe terms in negative samples. For example, the "organ" word strongly correlates with the "body" word, and if we just use "organ" to select negative samples, we shouldn't see many changes in the vibe difference compared to when we use both.

An example of this experiment is reported in Table 5. For this example, first, we get the difference between Tuned-VBERT and BM25 to find 15 high-impact terms, i.e., $\mathcal{W}_n$ in Eq.9: {"unit", "state", "popul", "includ", "censu", "answer", "time", "citi", "counti", "2010", "town", "area", "languag", "cell", "seat"}. We use these terms to find the vibe examples, i.e., $\mathcal{D}_{Vibe}$ in Eq.10 and train Tuned-VBERT-Neg with these negative samples. Then we compare Tuned-VBERT and Tuned-VBERT-Neg by Eqs.5 and 6.

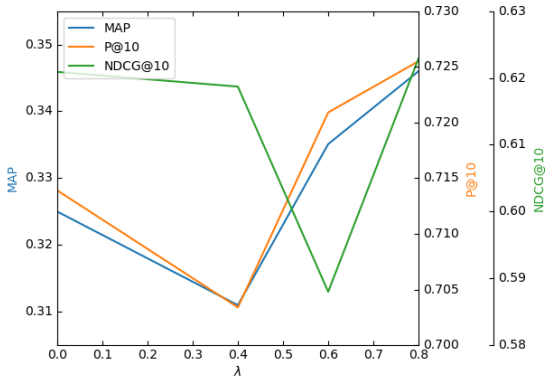

Figure 3: Effect of λ on validation set.

According to the Table 5, most of terms that we used in negative samples have disappeared in Tuned-VBERT-Neg, i.e., {"popul", "includ", "answer", "time", "citi", "counti", "town"}. This shows that Tuned-VBERT-Neg has less vibe than these words, and we are able to control the vibe in a model by the negative sampling strategy. The only word in the list that still appears in the high-impact terms of Tuned-VBERT-Neg compared to Tuned-VBERT is "cell". The reason may be related to the presence of this term in positive examples.

5.1.3 ***Analysis of the percentages of documents selected from the vibe documents.*** We evaluate our approach with different values for the percentages of documents from the vibe documents, i.e., λ in negative sampling. Figure 3 shows this experiment's results. When λ is equal to 0, it means that we *randomly* select all negative samples in training. By increasing λ to 0.4, we see the performance drop in all metrics, but when we set the value of λ to 0.6 and 0.8, the performance improves. This shows that the best value of λ is 0.8, so we need to select 80 percent of negative samples from vibe documents and 20 percent of them randomly.

5.2 Experiment for comparing two proposed approaches for detecting vibe

In this section, we aim to compare the two first proposed approaches for detecting vibe differences between two rankers. Note that approach three can just find the high-impact terms in the difference between the two rankers. However, the two first approaches can provide a quantity as the vibe metric. Therefore, in this section, we just compare these two approaches in terms of vibe metric.

We use the test set from the MS MARCO passage dataset consisting of 200 queries to compare these two approaches. In this experiment, we build two relevance-based language models for relevant (Rel) and non-relevant (Non-Rel) documents since we need to compare the distribution of each model with Rel and Non-Rel language models by using the proposed approaches. We use the test set instead of the dev set in this experiment since we for the test set we have more relevant documents for each query (9, 260 relevant documents for 200 queries). We also use two publicly available neural rankers we selected in section 3.3.1, i.e., VBERT and MonoT5.

The results of this experiments are reported in Table 6 and 7. According to Table 7, the vibe difference computed by approach two between MonoT5 and Rel language models (1.4566) is less than VBERT and Rel (1.5041). However, we see the opposite result in Table 6. Since we already know that the performance of the MonoT5 is better than VBERT, we expect that the vibe difference between MonoT5 and Rel language models be less than VBERT and Rel. So, approach two might be better for computing the vibe metric. A similar conclusion can be drawn considering the Non-Rel language model.

On the other hand, in our experiments, we find that approach one penalized terms that happened in both models. Therefore, it might be a better approach for finding terms that are significantly different in both models, which is suitable for detecting the vibe difference. We leave further comparisons between these two models for future work.

Table 6: The vibe metric computed by approach one, i.e., KL-Agg, between four language models.

	VBERT	MonoT5	Rel	Non-Rel
VBERT	0	1.0264	1.2016	3.6913
MonoT5	0.9381	0	1.2391	3.6900
Rel	1.2050	1.1867	0	3.7699
Non-Rel	6.1695	6.2734	6.0073	0

Table 7: The vibe metric computed by approach two, i.e., Agg-KL between four language models.

	VBERT	MonoT5	Rel	Non-Rel
VBERT	0	0.6881	1.5041	4.5858
MonoT5	0.7700	0	1.4566	4.7509
Rel	1.6606	1.7077	0	4.5861
Non-Rel	3.6432	3.6032	3.6279	0

6 CONCLUSIONS AND FUTURE WORK

In this paper, we define the *vibe* of a retrieval system as a relative comparison to a baseline search system. We proposed three approaches for detecting the vibe difference between the two ranking models. We studied three case studies: 1) comparing two publicly available rankers on general queries, 2) adding a vibe to a ranker, and 3) comparing click-based and judgment-based rankers. Finally, we proposed a practical approach to correct the detected vibe in a ranker based on negative sampling. We showed that the proposed approach for correcting the vibe could decrease the vibe and, at the same time, increase the permanence of the system for general queries.

An interesting future work could be answering this question: "How to find a subset of negative samples to maximize the evaluation measures, e.g., MAP or P@10?". Finding the correlation between the vibe metric and other evaluation measures can be an interesting future work. In this paper, we consider terms for detecting and correcting the vibe. However, using phrases instead of terms would be a more accurate approach.

ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

REFERENCES

[1] Rediet Abebe, Jon Kleinberg, and David Parkes. 2016. Fair division via social comparison. *arXiv preprint arXiv:1611.06589* (2016).

[2] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating position bias without intrusive interventions. In *WSDM'19*. 474–482.

[3] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W Bruce Croft. 2018. Unbiased learning to rank with unbiased propensity estimation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 385–394.

[4] Mozhdeh Ariannezhad, Ali Montazeralghaem, Hamed Zamani, and Azadeh Shakery. 2017. Improving retrieval performance for verbose queries via axiomatic analysis of term discrimination heuristic. In *SIGIR'17*. 1201–1204.

[5] Mozhdeh Ariannezhad, Ali Montazeralghaem, Hamed Zamani, and Azadeh Shakery. 2017. Iterative estimation of document relevance score for pseudo-relevance feedback. In *ECIR'17*. Springer, 676–683.

[6] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268* (2016).

[7] Amin Bigdeli, Negar Arabzadeh, Shirin Seyedsalehi, Morteza Zihayat, and Ebrahim Bagheri. 2022. A Light-Weight Strategy for Restraining Gender Biases in Neural Rankers. In *European Conference on Information Retrieval*. Springer, 47–55.

[8] L Elisa Celis, Damian Straszak, and Nisheeth K Vishnoi. 2017. Ranking with fairness constraints. *arXiv preprint arXiv:1704.06840* (2017).

[9] Le Chen, Ruijun Ma, Anikó Hannák, and Christo Wilson. 2018. Investigating the impact of gender on rank in resume search engines. In *Proceedings of the 2018 chi conference on human factors in computing systems*. 1–14.

[10] 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.

[11] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*. 87–94.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[13] Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. 67–73.

[14] Georges E Dupret and Benjamin Piwowarski. 2008. A user browsing model to predict search engine click data from past observations.. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. 331–338.

[15] Alessandro Fabris, Alberto Purpura, Gianmaria Silvello, and Gian Antonio Susto. 2020. Gender stereotype reinforcement: Measuring the gender bias conveyed by ranking algorithms. *Information Processing & Management* 57, 6 (2020), 102377.

[16] Hui Fang, Tao Tao, and ChengXiang Zhai. 2004. A formal study of information retrieval heuristics. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. 49–56.

[17] Luyu Gao and Jamie Callan. 2021. Unsupervised corpus aware language model pre-training for dense passage retrieval. *arXiv preprint arXiv:2108.05540* (2021).

[18] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. COIL: Revisit exact lexical match in information retrieval with contextualized inverted list. *arXiv preprint arXiv:2104.07186* (2021).

[19] Sahaj Garg, Vincent Perot, Nicole Limtiaco, Ankur Taly, Ed H Chi, and Alex Beutel. 2019. Counterfactual fairness in text classification through robustness. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 219–226.

[20] 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.

[21] Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2020. Learning-to-Rank with BERT in TF-Ranking. *arXiv preprint arXiv:2004.08476* (2020).

[22] Donna K Harman et al. 2005. The TREC test collections. (2005).

[23] Ayyoob Imani, Amir Vakili, Ali Montazer, and Azadeh Shakery. 2019. Deep neural networks for query expansion using word embeddings. In *ECIR'19*. Springer, 203–210.

[24] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *TOIS'02* 20, 4 (2002), 422–446.

[25] Thorsten Joachims and Filip Radlinski. 2007. Search engines that learn from implicit feedback. *Computer* 40, 8 (2007), 34–40.

[26] 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.

[27] Victor Lavrenko and W Bruce Croft. 2017. Relevance-based language models. In *ACM SIGIR Forum*, Vol. 51. ACM New York, NY, USA, 260–267.

[28] Craig Macdonald and Nicola Tonellotto. 2021. On approximate nearest neighbour selection for multi-stage dense retrieval. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3318–3322.

[29] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)* 54, 6 (2021), 1–35.

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *NIPS'13* 26 (2013).

[31] Ali Montazeralghaem, Razieh Rahimi, and James Allan. 2020. Relevance Ranking Based on Query-Aware Context Analysis. *ECIR'20* 12035 (2020), 446.

[32] Ali Montazeralghaem, Hamed Zamani, and James Allan. 2020. A reinforcement learning framework for relevance feedback. In *SIGIR'20*. 59–68.

[33] Ali Montazeralghaem, Hamed Zamani, and Azadeh Shakery. 2016. Axiomatic analysis for improving the log-logistic feedback model. In *SIGIR'16*. 765–768.

[34] Ali Montazeralghaem, Hamed Zamani, and Azadeh Shakery. 2017. Term proximity constraints for pseudo-relevance feedback. In *SIGIR'17*. 1085–1088.

[35] Ali Montazeralghaem, Hamed Zamani, and Azadeh Shakery. 2018. Theoretical analysis of interdependent constraints in pseudo-relevance feedback. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 1249–1252.

[36] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713* (2020).

[37] Jahna Otterbacher, Alessandro Checco, Gianluca Demartini, and Paul Clough. 2018. Investigating user perception of gender bias in image search: the role of sexism. In *The 41st International ACM SIGIR conference on research & development in information retrieval*. 933–936.

[38] Razieh Rahimi, Ali Montazeralghaem, and James Allan. 2019. Listwise neural ranking models. In *ICTIR'19*. 101–104.

[39] Navid Rekabsaz, Simone Kopeinik, and Markus Schedl. 2021. Societal biases in retrieved contents: Measurement framework and adversarial mitigation of bert rankers. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 306–316.

[40] Navid Rekabsaz and Markus Schedl. 2020. Do neural ranking models intensify gender bias?. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2065–2068.

[41] Justyna Sarzynska-Wawer, Aleksander Wawer, Aleksandra Pawlak, Julia Szymanowska, Izabela Stefaniak, Michal Jarkiewicz, and Lukasz Okruszek. 2021. Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research* 304 (2021), 114135.

[42] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of exposure in rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2219–2228.

[43] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 115–124.

[44] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 610–618.

[45] Ryen White. 2013. Beliefs and biases in web search. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. 3–12.

[46] Ke Yang and Julia Stoyanovich. 2017. Measuring fairness in ranked outputs. In *Proceedings of the 29th international conference on scientific and statistical database management*. 1–6.

[47] 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.

[48] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. 2017. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th international conference on world wide web*. 1171–1180.

[49] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. Fa* ir: A fair top-k ranking algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1569–1578.

[50] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *International conference on machine learning*. PMLR, 325–333.

[51] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. *arXiv preprint arXiv:1904.03310* (2019).

Appendices

A EXPERIMENTS

A.1 Comparing click-based and judgment-based rankers using approach three (TF Competition)

In this experiment, we aim to compare two proprietary rankers in Web search that were trained on two different types of data, i.e., clicks and judgments using the proposed approach three (see section 3.2.3). Approach three is to treat each query as a competition, where a higher term frequency (TF) wins. For example, for the word "gov" there were 74 queries where the TF was higher in the click-based ranker and 247 queries where the TF was higher in the judgment-based ranker. If it has TF= 0 on one side and TF> 0 on the other, we count it as a win for the one with some occurrence. The results of this experiment are reported in Table 8. Similar to the results of approach one (see section 3.3.3), we can see the word "quizlet" get a higher score for click-based ranker than judgment-based ranker.

Our observations from these results are as follows:

- **Authoritative sites are preferred by judges:** Authoritative sites like "gove", "edu", "nih", "nbci", and "nlm" are preferred by judges since they are reliable sites.
- **Sites that require the login penalized by judges**: Similar to the results of approach one, we see sites that require the login before using their content are penalized by judges.
- **Sites that require technical background penalized by judges**: sites like Stackoverflow, which is a question-and-answer website for professional and enthusiast programmers, are penalized by judges since they are trying to judge programming queries but have no technical background, so they don't understand the query or which sites are good.
- **Less authoritative sites are preferred in clicks:** Similar to the results of approach one, the forums and less authoritative sites are getting more clicks from users.

Table 8: Comparing click-based and judgment-based rankers using approach three (TF Competition).

Word	Click	Judgment	% Click
gov	74	247	23%
imdb	10	97	9%
title	22	105	17%
quizlet	40	1	98%
nih	4	41	9%
flowers	6	45	12%
cards	52	10	84%
flashcards	35	3	92%
net	191	103	65%
us	282	417	40%
store	97	157	33%
hl	8	43	16%
2021	431	302	59%
see	160	256	38%
gl	8	40	17%
details	80	151	35%
sur	33	5	87%
forum	55	17	76%
threads	34	6	85%
blogspot	36	7	84%
profile	29	74	28%
nlm	4	29	12%
ncbi	4	29	29%
stackoverflow	23	2	92%