A Counterfactual Explanation Framework for Retrieval Models

Anonymous ACL submission

Abstract

Explainability has become a crucial concern in today's world, aiming to enhance transparency in machine learning and deep learning models. Information retrieval is no exception to this trend. In existing literature on explainability of information retrieval, the emphasis has predominantly been on illustrating the concept of relevance concerning a retrieval model. The questions addressed include why a document is relevant to a query, why one document exhibits higher relevance than another, or why a specific set of documents is deemed relevant for a query.

However, limited attention has been given to understanding why a particular document is not favored (e.g. not within top-K) with respect to a query and a retrieval model. In an effort to address this gap, our work focus on the question of what terms need to be added within a document to improve its ranking. This in turn answers the question of which words played a role in not being favored in the document by a retrieval model for a particular query. We use a counterfactual framework to solve the above-mentioned research problem. To the best of our knowledge, we mark the first attempt to tackle this specific counterfactual problem (i.e. examining the absence of which words can affect the ranking of a document). Our experiments show the effectiveness of our proposed approach in predicting counterfactuals for both statistical (e.g. BM25) and deep-learning-based models (e.g. DRMM, DSSM, ColBERT, MonoT5). The code implementation of our proposed approach is available in https://anonymous.4open.science/r/CfIR-v2.

1 Introduction

The requirement of transparency of AI models has made explainability crucial, and this applies to Information Retrieval (IR) models as well (Anand et al., 2022). The target audience plays a significant role in achieving explainability for an information retrieval model, as the units of explanation or questions may differ based on the end user. For instance, a healthcare specialist, who is a domain expert but not necessarily an information retrieval specialist, might want to understand the reasons behind a ranked suggestion produced by a retrieval model in terms of words used (Singh and Anand, 2019). On the other hand, an IR practitioner may be more interested in understanding whether different IR axioms are followed by a retrieval model or not (Bondarenko et al., 2022).

This study focuses on the perspective of Information Retrieval (IR) practitioners. To be more specific, we introduce a counterfactual framework designed for information retrieval models, catering to the needs of IR practitioners. Existing literature in explainable IR (ExIR) addresses questions like why a particular document is relevant with respect to a query (Singh and Anand, 2019), between a pair of documents why one document is more relevant to the query (Penha et al., 2022) compared to the other and why a list of documents relevant to a query (Lyu and Anand, 2023). Broadly speaking, all the above-mentioned questions mainly focus on explaining the relevance of a document or a list of documents from different perspectives.

However, there is limited attention to explain the question like the absence of which words renders a document unfavorable to a retrieval model (i.e. not within top-K) remains unexplored. The above-mentioned explanation can give an idea to an IR practitioner about how to modify a retrieval model. For example, if it is observed that a retrieval model (e.g. especially neural network based retrieval models) does not favor documents because of not having words which are not so related to query topic then the setting of the retrieval model needs to be changed so that it gives more

importance to the semantic similarity feature.

With the motivation described above, the fundamental research question which we address in this research work is described as follows.

- **RQ1:** What are the terms that should be added to a document which can push the document to a higher rank with respect to a particular retrieval model?

We would like to note that we have framed **RQ1** as a counterfactual setup in our research scope. Similar to existing research in counterfactual explanations in AI (Kanamori et al., 2021; Van Looveren and Klaise, 2021), we also attempt to change the output of model with the provided explanations (i.e. change the rank of a document in IR models). Our experimental results show that on an average in 70% cases the solution provided by the counterfactual setup improves the ranking of a document with respect to a query and a ranking model.

Our Contributions The main contributions of this paper are as follows.

- Propose a model-agnostic novel counterfactual framework for retrieval models.
- Estimated a set of terms that can explain why a document is not within top-K with respect to a query and a retrieval model.
- Provide a comprehensive analysis with existing state-of-the-art IR models.

The rest of the paper is organized as follows. Section 2 describes Related work. Section 3 describes the counterfactual framework used in our work, Section 4 describes the experimental setup and Section 5 discuss about results and ablation study. Section 6 concludes with this paper.

2 Related Work

Existing research related to this work can be broadly categorized into three different areas: **a)** Counterfactual Explainability in general AI, **b)** Explainability in IR and **c)** Search engine optimization. Each method are described as follows.

2.1 Counterfactual Explanations

The xAI field gained significant momentum with the development of the Local Interpretable Model-agnostic Explanations (LIME) method (Ribeiro et al., 2016), which offers a way to explain any classification model. While models like LIME explain why a model predicts a particular output, counterfactual explainers address the question of what changes in input features would be needed to alter the output. Counterfactual xAI was first brought into the limelight in early 2010s with seminal works of Judea Pearl (2018). Karimi et al. (2020) provided a practical framework named Model-Agnostic Counterfactual Explanations (MACE) for generating counterfactual explanations for any model. Later series of models (Kanamori et al., 2021; Van Looveren and Klaise, 2021; Parmentier and Vidal, 2021; Carreira-Perpiñán and Hada, 2021; Pawelczyk et al., 2022; Hamman et al., 2023) based on optimization framework were proposed for counterfactual explanation. In our research scope, we use Counterfactual Explanation framework proposed in (Mothilal et al., 2020).

2.2 Explainability in IR

Pointwise Explanations Here the explainer shows the important features responsible for the relevance score predicted by a retrieval model for a query-document pair. Popular techniques include locally approximating the relevance scores predicted by the retrieval model using a regression model (Singh and Anand, 2019).

Pairwise Explanations Here explainers predict why a particular document was favored by a ranking model compared to others. The work in (Xu et al., 2024) proposed a counterfactual explanation method to compare the ranking of a pair of documents with respect to a particular query.

Listwise Explanations Here the focus is on explaining the key features for a ranked list of documents and a query. Listwise explanations (Yu et al., 2022; Lyu and Anand, 2023) aim to capture a more global perspective compared to pointwise and pairwise explanations. The study in (Lyu and Anand, 2023) proposed an approach which combines the output of different explainers to capture the different aspects of relevance. The study in (Yu et al., 2022) trained a transformer model to generate explanation terms for a query and a ranked list of documents.

Generative Explanation Unlike previously mentioned methods, which focus on analyzing existing features or model internals, generative explanations (Singh and Anand, 2020; Lyu and Anand, 2023) leverage natural language process-

ing to create new text content, like summaries or justifications, that directly address the user's query and information needs. Model-agnostic approaches (Singh and Anand, 2020) have been proposed to interpret the intent of the query as understood by a black box ranker.

From the above mentioned category of explanations in IR, we focus on pointwise explanation in our research scope. In pointwise explanation, rather than explaining what are the words which are relevant in a document for a particular query we address the the research quetion what are the words which are required to improve the ranking of the document with respect to a query.

Search Engine Optimization The study in (Egri and Bayrak, 2014; Erdmann et al., 2022) uses different features like commercial cost, links to optimize the performance of the search engine. A major difference of the work in (Egri and Bayrak, 2014; Erdmann et al., 2022) with our work is we only consider the words present in a document as a feature. Our objective is to improve the ranking of a particular document concerning a specific query and a retrieval model rather than improving the ranking of a document concerning any query belonging to a particular topic.

3 Counterfactual Framework for Information Retrieval (CFIR)

In this section, we first outline the counterfactual setup proposed in (Mothilal et al., 2020), followed by a detailed explanation of the counterfactual setup in IR. The work in (Mothilal et al., 2020) focused on identifying counterfactuals in regression and classification scenarios. Primarily the research question addressed in (Mothilal et al., 2020), is identifying which features in the input instance need to be modified to change the output of a trained model.

Counterfactual Setup (CF Setup) The counterfactual generator described in (Mothilal et al., 2020), takes as input a trained machine learning model f (generally a classification or regression model), an instance, x for which we want to generate counterfactual examples and the number of different counterfactual examples that need to be generated k and the output is k number of counterfactuals (denoted as $\{c_1, c_2 \ldots, c_k\}$) for x. Each c_i is designed to alter the prediction for x in f. The main assumptions in the above-mentioned setup are that the machine learning model (i.e. f) should be differentiable and the output of the model should not change over time.

The loss function for the counterfactual generator tries to minimize three different criteria to generate counterfactuals. They are

- **Criteria 1:** Minimizing the distance between the desired outcome y' and the prediction of the model f for a counterfactual example $(f(c_i))$.
- **Criteria 2:** Minimizing the distance between the generated counterfactual (c_i) and the original input x. Broadly speaking, a counterfactual example closer to the original input should be more useful for a user.
- **Criteria 3:** Increasing diversity between generated counterfactuals.

Based on the above-mentioned criteria the loss function for the counterfactual generator $(C(f, x, k))$ is described as follows.

$$\begin{aligned} C(f, x, k) &= \{c_1, c_2 \ldots c_k\} \\ &= \underset{c_1,\ldots c_k}{\arg\min} \left(\frac{1}{k} \sum_{i=1}^{k} \text{yloss}(f(c_i), y) + \right. \\ &\left. \frac{\lambda_1}{k} \sum_{i=1}^{k} \text{dist}(c_i, x) - \lambda_2 \text{div}(c_1, \ldots, c_k) \right) \end{aligned} \tag{1}$$

Equation 1 essentially finds a set k number of c_is for which the sum of all the three criteria is minimized. In Equation 1, yloss(.) takes care of the first criterion, dist(c_i, x) takes care of the second criterion and div takes care of the third criterion as discussed above. λ_1 and λ_2 in Equation 1 are hyperparameters that balance the contribution of second and third parts of loss function (i.e. controlling diversity and feasibility). The detailed description of the computation of $yloss$, $dist$ and div function in Equation 1 is given in Equations 5, 6 and 7 respectively in Appendix 9.4. The loss function in Equation 1 is optimized using the gradient descent method.

3.1 Mapping Retrieval to CF Setup

In IR, the end user is generally interested in the ranking of documents within top-K. Hence to align with the counterfactual setup (specifically input f in Equation 1) described in Section 3, we aim to build on a classifier where we are interested in finding the counterfactuals that can push a document within top-K. The specifics of the classification setup are given below.

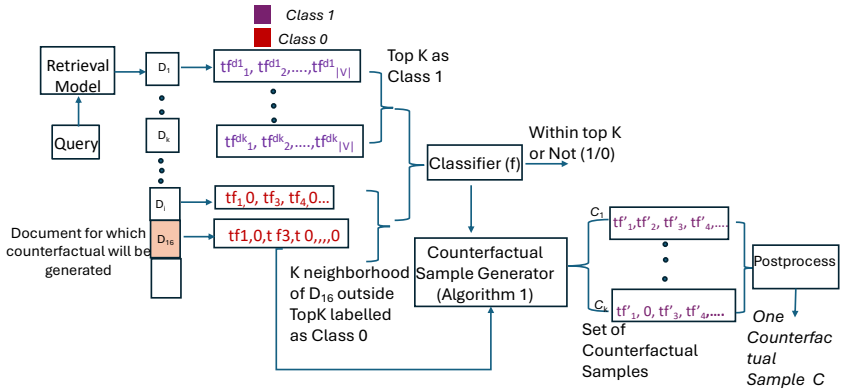


Figure 1: Counterfactual Explanation Model Description

Classifier Setup Existing work in xAI developed a simple model which can approximate the decision boundary of the original complex model in a small region (Ribeiro et al., 2016) to explain the original model. In that direction, the objective of the classifier in our research scope is to locally approximate the behavior of a retrieval model M, for a query q and a subset of documents retrieved for the query q. More specifically, we are interested in approximating the behavior of the retrieval model M in determining whether a document is retrieved within the top-K results or not. In contrast to (Ribeiro et al., 2016) we build a binary classification model instead of a regression model. The binary classifier predicts whether a document d will be ranked within the top-K results or not for a specific query q and retrieval model M.

For each document d for which we want to generate counterfactuals, we train separate classifier. In the classifier setup, the top-K documents for a query q and model M represent class 1 and any other document not belonging to this class represents class 0. Theoretically speaking, if a corpus had N number of documents, then there will be $N-K$ documents which should have class label 0 and $N-K$ is a very large number in general which can cause class imbalance issue. To avoid this issue, for the 0 class, for each document d for which we want to generate a counterfactual, we choose a set of closest neighbors in the set of $N-K$ documents and the size of the neighborhood should be similar to K. Consequently, there is no class imbalance issue in our classifier setup. In the classifier setup, K serves as a predefined threshold, typically set to values such as 10, 20, or 30.

Feature Vector for Classifier The classifier function (i.e. the instance of f in Equation 1) takes as input a word based feature vector corresponding to a document (i.e. the instance of x in Equation 1). Generating the feature vector for the classifier using all the words from documents retrieved for a query can pose challenges. Consequently, we adopted a filtering strategy, where we selectively choose the most significant n words from each document d using a function named $Imp(d)$. We create a vocabulary set V by taking the union of the top n important words present in each of the top-K documents of the ranked list. Mathematically, $V = \cup_{i=1}^{K}\{\sum_{j=1,w_j \in Imp(d_i)}^{n} w_j\}$. We explored three different mechanisms (described in Section 4) to implement $Imp(d)$. The dimension of the feature vector required for the classifier setup is set to the size of the vocabulary set (i.e. $|V|$), where each position within the feature vector maps to a unique word in the set V. The feature vector representation of d is represented as $d_{vec} = \{tf_1^d, tf_2^d \ldots, tf_{|V|}^d\}$ where tf_i^d represents the term frequency of the word w_i in d. Appendix 9.2 depicts a step-by-step algorithm to construct the feature vector for the classifier and Figure 6 in Appendix 9.2 shows one sample feature vector for the classifier. Equation 2, defines the instance of f (as given in Equation 1) in the retrieval setup.

$$f : R^{|V|} \rightarrow \{0, 1\} \tag{2}$$

In Equation 2, for each document d, f takes d_{vec} as input and predicts whether the class label as 0 or 1. Once the classifier is trained, we use this classifier to train the counterfactual model described in Equation 1 to generate the counterfactuals. Although Equation 1 generates k different counterfactuals, we perform post-processing (as described in step 5 in Algorithm 1) in the retrieval setup to eventually generate one counterfactual sample (i.e. c) which is a $|V|$ dimensional vector. Algorithm 1

shows step by step execution of the counterfactual explanation generator. Algorithm 1 shows how the potential counterfactual examples $(c_1, \ldots c_k)$ are randomly initialized. The counterfactual output in the retrieval setup (i.e. c) should change the classifier prediction from 0 to 1 or 1 to 0. In our research scope, we are primarily focused on the scenario where the classifier label is changed from 0 to 1 (i.e. modified document should be within top K). The set of words corresponding to the counterfactual explanation of d are the new words that have been added to d_{vec} in the output of Algorithm 1 (i.e. c). Mathematically, the set of counterfactual explanations ($CFEXP(d)$) for a document d can be written as follows.

$$CFEXP(d) = \cup_{j=1}^{|V|}\{\cup_{i=0}^{|c_j - tf_j^d|}\{w_j\}\} \quad (3)$$

In Equation 3 c_j is term frequency of w_j in the counterfactual vector c and tf_j^d is the term frequency of the word w_j in the original document d. Figure 1 shows the schematic diagram for counterfactual setup with the workflow between the different components (i.e. classifier and counterfactual generator) within it.

Algorithm 1: CF Explanation Generator

Input : Classifier function: f, Feature Vector: $d_{vec} = \{tf_1, tf_2, \ldots, tf_{|V|}\}$, Number of Counterfactuals:k
Output : $\{c \in R^{|V|}\}$
Initialization:
  for $i \leftarrow 1$ **to** k **do**
    for $j \leftarrow 1$ **to** $|V|$ **do**
      $c_{i,j}^0 = r \sim Random(.)$
      /* $c_{i,j}^0$ is the j^{th} coordinate of c_i at 0^{th} iteration */
    end for
  end for
for $t \leftarrow 0$ **to** $maxIter$ **do**
  Compute the loss $\frac{1}{k}\sum_{i=1}^{k} \text{yloss}(f(c_i^t), y) + \frac{\lambda_1}{k}\sum_{i=1}^{k} dist(c_i^t, x) - \lambda_2 div(c_1^t, \ldots, c_k^t))$
  Update c_i^t's using gradient descent
end for
return c c is a $|V|$ dimensional vector randomly chosen from the subset of $c_i^{maxIter}$'s for which $c_{i,j}^{maxIter} \geq tf_j^d \; \forall j = 1, \ldots, |V|$

4 Experiment Setup

Dataset We use three ranking datasets for our experiments: MS MARCO passage dataset for short documents (Bajaj et al., 2016) and MS MARCO document ranking dataset for longer documents (Craswell et al., 2023) and TREC Robust (Voorhees, 2005) dataset. The MS MARCO passage and document ranking datasets contain queries from Bing[1] and the queries of TREC Robust are manually chosen. For each dataset, we randomly selected 100 queries from the test set and chose 5 documents not ranked in the top 10 results for each query, resulting in a test set of 250 query-document pairs.The details of the dataset are given in Table 1.

		MS MARCO Passage	MS MARCO Document	TREC Robust
Query	Avg Length	5.9	6.9	7.18
Document	Avg Length	64.9	1134.2	150.12
Query	#Instances	100	1000	100
Document	#Instances	250	250	250

Table 1: Dataset Details for Counterfactual Setup

Retrieval Models The five different retrieval models used in our experiment are described as follows.

BM25: BM25[2] is a statistical retrieval model where the similarity between a query and a document is computed based on the term frequency of the query words present in the document, document frequency of the query words and also the document length.

DRMM: Deep Relevance Matching Model (DRMM) Guo et al. (2016) is a neural retrieval model where the semantic similarity between each pair of tokens corresponding to a query and a document is computed to estimate the final relevance score of a document.

DSSM: Deep Semantic Similarity Model (DSSM) Huang et al. (2013) is another neural retrieval model which uses word hashing techniques to compute the semantic similarity between a query and a document.

ColBERT: Contextualized Late Interaction over BERT (ColBERT) (Khattab and Zaharia, 2020), is an advanced neural retrieval model which exploits late interaction techniques based on BERT (Devlin et al., 2019) based representations of both query and document for retrieval.

MonoT5: MonoT5 (Nogueira et al., 2020) is a sequence-to-sequence model fine-tuned to predict the relevance of a query-document pair.

Baselines To the best of our knowledge, this is the first work which attempts to provide counterfactual explanations in IR. Consequently, there exists no baseline for our proposed approach. However we have used a query word and top-K word

[1] https://bing.com
[2] https://en.wikipedia.org/wiki/Okapi_BM25

based intuitive baseline to compare with our proposed approach. In query word baseline (QW), we use query words not originally present in a document to enhance its ranking. For Top-K' ($Top-K'$) baseline we use the top k' words extracted from top 5 documents corresponding to a query as relevance set. Words appearing in the relevance set but not appearing in a document are added to the document to improve its ranking. For different retrieval models we have corresponding versions of QW and $Top-K'$ baselines.

Evaluation Metrics There exists no standard evaluation framework for exIR approaches. The three different evaluation metrics in our experiment setup are described as follows.

Fidelity (FD): Existing xAI approaches in IR use Fidelity (Anand et al., 2022) as one of the metrics to evaluate the effectiveness of the proposed explainability approach. Intuitively speaking, Fidelity measures the correctness of the features obtained from a xAI approach. In the context of the CFIR setup described in this work, we define this fidelity score as the number of times the words predicted by the counterfactual algorithm could actually improve the rank of a document. Let n be total number of query document pairs in our test case and x be number of query document pairs for which the the rank of the document improved after adding the counterfactuals obtained from the optimization setup described in Equation 1. Then the Fidelity score is mathematically defined with respect to a test dataset D and retrieval model M is defined as follows.

$$FD(D, M) = \frac{x}{n} * 100 \tag{4}$$

Avg. New Words: Here we compute the average number of new words added by the counterfactual approach for a set of query document pairs.

Avg. Query Overlap: Here we report on an average how many of the words suggested by the counterfactual algorithm come from the query words.

Parameters and Implementation Details The details of implementation about retrieval models are shown in Appendix 9.1. We employed two popular classical machine learning methods Logistic Regression (LR) and Random Forest (RF) for the classifier described in Section 3.1. For Logistic Regression the learning rate was set to 0.001. For Random Forest the number of estimators were set to 100. We train a separate classifier for each query and retrieval model. In total, for each retrieval model there are 100 classifiers. As described in Section 3.1, all the words present in a document is not used as input to the classifier. We use top 10 ($n' = 10$) most important words from a document. As described in Section 3.1, we explored three different ways to implement $Imp(d)$ function a) TF-IDF weight based word extraction, b) BERT based keyword extraction (Grootendorst, 2020) and c) Similarity between the BERT representation of query and the document tokens. We found that BERT representation based similarity computation worked the best for our approach. More details on the implementation of $Imp(d)$ function are shown in Appendix 9.7. Then top n words from each document are used to create the vocabulary ($|V|$) for the classifier. More details on the parameter configuration are shown in Appendix 9.5.

5 Results

Table 2 shows the performance of the counterfactual approach across different retrieval models (i.e. BM25, DRMM, DSSM, ColBERT, MonoT5). We conducted experiments on MS MARCO passage document and TREC Robust dataset to observe the effectiveness of our proposed explanation approach for different types of documents. Mainly four different observations can be made from Table 2. ***Firstly***, It can be clearly observed that the CFIR model for each retrieval model has performed better compared to its corresponding query word or top-K' words baseline in terms of Fidelity score(FD). The above-mentioned observation is consistent for both passages and long documents (i.e. both in MSMARCO passage and Document). ***Secondly***, it can be observed from Table that mostly CFIR approach provided the highest number of new terms (terms not already present in the documents) as part of the explanation to improve ranking. Consequently, we can say the overall set of explanation terms are more diverse for CFIR approach compared to others. It can also be also observed from Table 2 that the Fidelity scores are generally better in the MS MARCO passages compared to MSMARCO document and TREC Robust dataset. One likely explanation for this phenomenon is that documents in MSMARCO document and TREC Robust are longer in length

Model Description		MS MARCO Passage			MS MARCO Document			Trec Robust		
Retrieval Model	**Classifier**	**FD(%)**	**Avg. New Words**	**Avg. Query Overlap**	**FD(%)**	**Avg. New Words**	**Avg. Query Overlap**	**FD(%)**	**Avg. New Words**	**Avg. Query Overlap**
QW_{BM25}	NA	50%	5.61	100%	48%	6.14	100%	56%	6.12	100%
$Top-K'_{BM25}$	NA	42%	11.28	100%	40%	9.61	100%	41%	12.34	100%
$CFIR_{BM25}$	RF	65%	10.64	66%	52%	**16.81**	56%	**64%**	11.12	57%
$CFIR_{BM25}$	LR	**69%**	**17.14**	58%	**57%**	14.15	56%	58%	**13.25**	56%
QW_{DRMM}	NA	48%	5.12	100%	47%	6.14	100%	49%	7.12	100%
$Top-K'_{DRMM}$	NA	42%	**15.11**	100%	31%	14.12	100%	33%	**16.12**	100%
$CFIR_{DRMM}$	RF	**72%**	11.31	48%	56%	8.12	46%	62%	12.56	47%
$CFIR_{DRMM}$	LR	68%	12.37	62%	**62%**	**14.53**	45%	**65%**	13.47	43%
QW_{DSMM}	NA	49%	5.32	100%	45%	6.64	100%	52%	7.12	100%
$Top-K'_{DSSM}$	NA	35%	12.51	100%	32%	12.62	100%	34%	13.14	100%
$CFIR_{DSSM}$	RF	57%	11.52	58%	46%	18.14	57%	**59%**	12.46	100%
$CFIR_{DSSM}$	LR	**62%**	**15.78**	54%	**53%**	**18.52**	63%	58%	**17.24**	64%
$QW_{ColBERT}$	NA	56%	4.78	100%	34%	5.64	100%	38%	6.14	100%
$Top-K'_{ColBERT}$	NA	48%	**15.63**	100%	36%	**13.42**	100%	38%	11.32	100%
$CFIR_{ColBERT}$	RF	72%	12.41	56%	**72%**	11.05	49%	71%	10.35	52%
$CFIR_{ColBERT}$	LR	**75%**	14.12	61%	71%	10.23	62%	**74%**	**16.45**	65%
QW_{MonoT5}	NA	52%	10.15	100%	54%	12.23	100%	63%	10.15	100%
$Top-K'_{MonoT5}$	NA	75%	**14.11**	100%	68%	10.13	100%	75%	11.12	100%
$CFIR_{MonoT5}$	RF	80%	12.13	64%	72%	11.23	61%	73%	10.95	66%
$CFIR_{MonoT5}$	LR	**82%**	13.15	65%	**74%**	**12.23**	63%	**75%**	**11.45**	68%

Table 2: CFIR model Performance for BM25, DRMM, DSSM and ColBERT, MonoT5 in MSMARCO Passage and Document Collection and TREC Robust. The Best Performing Counterfactual Explanation Method for every retrieval model is boldfaced; the overall best performance across all rows is underlined

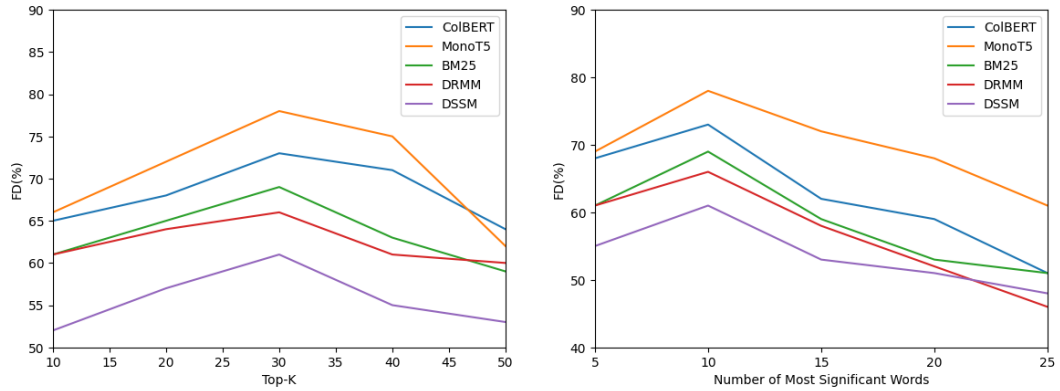
Retrieval Model	Query Text	docId	Explanation Terms
DRMM	What law repealed prohibition ?	3686955	working, strict, Maine, 1929, law, resentment, New York City, Irish, immigrant, prohibition, repeal, fall, Portland, temperance, riot, visit
DSSM	What is the role of lipid in the cell?	6159679	phospholipid, fluidity, storage, triglyceride, fatty receptor
ColBERT	what type of wave is electromagnetic?	5217641	directly ,oscillations, medium, wave, properties, speed

Table 3: Sample Explanation Terms by CFIR Model for DRMM, DSSM and ColBERT in MS MARCO passage

compared to passages. Consequently, it is easier for shorter documents to change the ranking compared to longer documents. ***Thirdly***, another interesting observation from Table 2 is that the maximum query word overlap by our proposed approach is 63%. This implies that the counterfactual algorithm is suggesting new words that are not even present in a query. ***Fourthly***, the performance of representation learning based retrieval models (i.e. ColBERT, MonoT5) are significantly better than the other models for Fidelity metric. One potential reason can be that, the counterfactual generator suggests words which are similar to the content of the document. Because of using better embedding representation (BERT (Devlin et al., 2019) and T5 compared to Word2Vec (Mikolov et al., 2013) in DRMM) these retrieval models give more priority to similar words than other retrieval models.

There exists work (Liu et al., 2023; Wu et al., 2022a) in IR that focused on adversarial attacks, where content or embedding of a document is modified to improve its ranking. The key distinction of our work from existing adversarial approaches is that, instead of altering the document encoding or replacing words, we focus solely on adding new words to the document. However, for comparison, we have evaluated the performance of CFIR against the PRADA (Wu et al., 2022a) model which replaces certain words in a document to improve its ranking. Table 8 in Appendix 9.6 shows that CFIR performs better than PRADA for both ColBERT and MonoT5 in terms of Fidelity score. Table 3 shows some example terms extracted by our proposed approach. The words shown in Table 3 have improved the ranking of a docID with respect to the queries shown.

Parameter Sensitivity Analysis In Table 2, we observed that for most of the retrieval models the performance of the counterfactual explainer follows similar trend both in MSMARCO passage and document dataset (i.e. the best performing model in terms of fidelity score is same in most of the cases). As a result, we conducted parameter sensitivity experiments only on MSMARCO passage dataset. Figure 2 (a) shows the variance in Fidelity score with respect to the K value in Top-K. In Figure 2 (b) we show the variance of FD score with respect to the number of most significant words (i.e. n) used to construct the document vector. It is clearly visible from Figure 2 (b) that with an increase in the number of counterfactuals, there is a decrease in the performance



(a) (b)

Figure 2: Counterfactual Classifier Performance Variance with Top-K and Counterfactual Performance Variance with variation of number of Counterfactuals

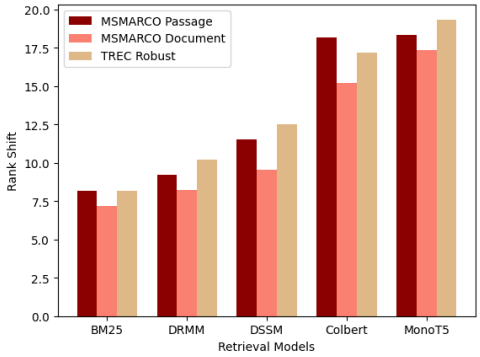


Figure 3: Average Rank shift by CFIR for BM25, DRMM, DSSM, ColBERT and MonoT5

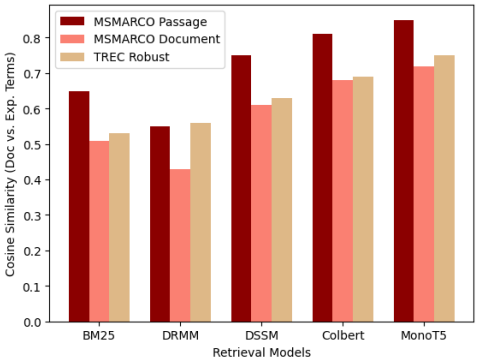


Figure 4: Average Semantic Similarity between original documents and the corresponding counterfactual explanation Terms for BM25, DRMM, DSSM, ColBERT and MonoT5

of the counterfactual classifier. It can be observed that for $n = 10$ the best performance is achieved. Intuitively, as the number of words increases, the feature vector grows exponentially, making it challenging to train the classifier effectively. Figure 3 shows the average change in rank after introducing the explanation terms suggested by the CFIR setup. Figure 3 essentially demonstrates the actionability introduced by the counterfactual explanation terms. The two things to observe from Figure 3 are firstly, the average rank shift is greater for documents than for passages. Table 2 shows that ColBERT achieved a significantly higher fidelity score (16^{th} row) and a larger average rank shift compared to the other models, as also seen in Figure 3. Figure 4 shows the average cosine similarity computed between documents and the corresponding explanation terms. For both documents and the explanation terms we use pretrained BERT representations to compute the similarity. It can be observed from Figure 4 that the cosine similarity for the representation learning based retrieval models (i.e. ColBERT, MonoT5) are higher than the other retrieval models in general.

6 Conclusion

In this paper, we propose a counterfactual setup for a query-document pair and a retrieval model. We conducted experiments on both MS MARCO passage and document ranking sets. Our experiments show that the proposed approach on an average 70% cases for both in short and long documents could successfully improve the ranking. In the future, we would like to explore different explanation units for the counterfactual setup.

7 Limitations

One of the limitations of this work is that we assume that top 10 or 20 words (based on tf-idf

weights) within a document play the most important part in improving the rank of a document. However, theoretically speaking we should consider all the words present in a document to determine the most influential words for a retrieval model. We have used top tf-idf words (Similar to statistical retrieval models) to reduce the computational complexity of our experiments and we have seen that increasing the number of top words doesn't affect the performance of the model that much.

8 Ethical Considerations

In this work, we have used publicly available search query log and document collection to demonstrate counterfactual explanation. No sensitive data was used in this experiment. As a result of this there is no particular ethical concern associated with this work. If there is any kind of bias present in the search log data that effect can be observed within our approach. However mitigating that bias was beyond the scope of this work

References

Avishek Anand, Lijun Lyu, Maximilian Idahl, Yumeng Wang, Jonas Wallat, and Zijian Zhang. 2022. Explainable information retrieval: A survey.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. Ms marco: A human generated machine reading comprehension dataset. In *InCoCo@NIPS*.

Alexander Bondarenko, Maik Fröbe, Jan Heinrich Reimer, Benno Stein, Michael Völske, and Matthias Hagen. 2022. Axiomatic retrieval experimentation with ir_axioms. In *Proc. of SIGIR 2022*, pages 3131–3140.

Miguel Á Carreira-Perpiñán and Suryabhan Singh Hada. 2021. Counterfactual explanations for oblique decision trees: Exact, efficient algorithms. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6903–6911.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the TREC 2020 deep learning track. *CoRR*, abs/2102.07662.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Jimmy Lin, Ellen M. Voorhees, and Ian Soboroff. 2023. Overview of the trec 2022 deep learning track. In *Text REtrieval Conference (TREC)*. NIST, TREC.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Gokhan Egri and Coskun Bayrak. 2014. The role of search engine optimization on keeping the user on the site. *Procedia Computer Science*, 36:335–342. Complex Adaptive Systems Philadelphia, PA November 3-5, 2014.

Anett Erdmann, Ramón Arilla, and José M. Ponzoa. 2022. Search engine optimization: The long-term strategy of keyword choice. *Journal of Business Research*, 144:650–662.

Maarten Grootendorst. 2020. Keybert: Minimal keyword extraction with bert.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, page 55–64, New York, NY, USA. Association for Computing Machinery.

Jiafeng Guo, Yixing Fan, Xiang Ji, and Xueqi Cheng. 2019. Matchzoo: A learning, practicing, and developing system for neural text matching. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pages 1297–1300.

Faisal Hamman, Erfaun Noorani, Saumitra Mishra, Daniele Magazzeni, and Sanghamitra Dutta. 2023. Robust counterfactual explanations for neural networks with probabilistic guarantees. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, CIKM '13, page 2333–2338.

Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, Yuichi Ike, Kento Uemura, and Hiroki Arimura. 2021. Ordered counterfactual explanation by mixed-integer linear optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11564–11574.

Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. 2020. Model-agnostic counterfactual explanations for consequential decisions. In *International Conference on Artificial Intelligence and Statistics*, pages 895–905. PMLR.

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*, page 39–48.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 2356–2362.

Yu-An Liu, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, Yixing Fan, and Xueqi Cheng. 2023. Black-box adversarial attacks against dense retrieval models: A multi-view contrastive learning method. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, page 1647–1656, New York, NY, USA. Association for Computing Machinery.

Lijun Lyu and Avishek Anand. 2023. Listwise explanations for ranking models using multiple explainers. In *Advances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2–6, 2023, Proceedings, Part I*, page 653–668, Berlin, Heidelberg. Springer-Verlag.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*.

Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pre-trained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718, Online. Association for Computational Linguistics.

Axel Parmentier and Thibaut Vidal. 2021. Optimal counterfactual explanations in tree ensembles. In *International conference on machine learning*, pages 8422–8431. PMLR.

Martin Pawelczyk, Chirag Agarwal, Shalmali Joshi, Sohini Upadhyay, and Himabindu Lakkaraju. 2022. Exploring counterfactual explanations through the lens of adversarial examples: A theoretical and empirical analysis. In *International Conference on Artificial Intelligence and Statistics*, pages 4574–4594. PMLR.

Judea Pearl. 2018. Theoretical impediments to machine learning with seven sparks from the causal revolution. *arXiv preprint arXiv:1801.04016*.

Gustavo Penha, Eyal Krikon, and Vanessa Murdock. 2022. Pairwise review-based explanations for voice product search. In *ACM SIGIR Conference on Human Information Interaction and Retrieval*, pages 300–304.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?": Explaining the predictions of any classifier. In *Proc.of SIGKDD 2016*, page 1135–1144.

Jaspreet Singh and Avishek Anand. 2019. Exs: Explainable search using local model agnostic interpretability. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, page 770–773.

Jaspreet Singh and Avishek Anand. 2020. Model agnostic interpretability of rankers via intent modelling. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 618–628.

Arnaud Van Looveren and Janis Klaise. 2021. Interpretable counterfactual explanations guided by prototypes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 650–665. Springer.

Ellen Voorhees. 2005. Overview of the trec 2004 robust retrieval track.

Chen Wu, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2022a. Prada: Practical black-box adversarial attacks against neural ranking models. *ArXiv preprint*, abs/2204.01321.

Chen Wu, Ruqing Zhang, Jiafeng Guo, Yixing Fan, and Xueqi Cheng. 2022b. Are neural ranking models robust? *ACM Trans. Inf. Syst.*, 41(2).

Zhichao Xu, Hemank Lamba, Qingyao Ai, Joel Tetreault, and Alex Jaimes. 2024. Counterfactual editing for search result explanation. *Preprint*, arXiv:2301.10389.

Puxuan Yu, Razieh Rahimi, and James Allan. 2022. Towards explainable search results: a listwise explanation generator. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 669–680.

9 Appendix

9.1 Retrieval Performance of IR Models

We use Lin et al. (2021) toolkit for implementing BM25 and MonoT5. For DRMM and DSSM, we use the implementation released by the study in Guo et al. (2019). For passage ranking we varied the parameters in a grid search and we took the configuration producing best MRR@10 value on TREC DL (Craswell et al., 2021) test set. For both DRMM and DSSM experiments on MSMARCO data, the parameters were set as suggested in (Wu et al., 2022b). The MRR@10 values are reported in Table 7 in Appendix 9.1.

For DRMM and DSSM, we use randomly chosen $100K$ query pairs from the MSMARCO training dataset to train the model.

	MRR@10	
Model	**MSMARCO Passage**	**MSMARCO Document**
BM25	0.1874	0.2184
DRMM	0.1623	0.1168
DSSM	0.1320	0.1168
ColBERT	0.3481	0.3469
MonoT5	0.3904	0.3827

Table 4: Retrieval Model Performance on MSMARCO passage and document

9.2 Example of Input and Output to Classifier

Given an input query, we employ a LuceneSearcher with MSMARCO Index to retrieve the Top-K documents. The feature vector construction process follows these steps:

For each document, we:

1. Extract the top n words based on their Imp(d) values
2. Construct a vocabulary V as the union of all top 10 words across documents
3. Note that $|V|$ typically falls in the range of 150-180 words

The feature vector for each document has dimension $|V|$, where each component represents the value from the Imp(d) of the corresponding word from the vocabulary. Formally:

$$\mathrm{d}_{vec}\mathrm{d}_{vec}\mathrm{d}_{vec\mathrm{d}_{vec}} \in R^{|V|}$$

Labels are assigned according to the following criterion:

$$\text{label} = \begin{cases} 1 & \text{for top } K \text{ documents} \\ 0 & \text{for remaining documents} \end{cases}$$

Example feature vectors and their corresponding counterfactuals generated using (Mothilal et al., 2020) are shown in Table 6. Since |V| is 150 in our experiments, hence in Table 6 we have only shown the term frequencies of the words present in each document. For other words the terms freaquency values will be zero in d_{vec}.

Existing Explanation Methods	**Word Overlap**
PointWise Explanation (Singh and Anand, 2019)	21.46%
ListWise Explanation (Lyu and Anand, 2023)	9.57%

Table 5: Comparison of CFIR with Existing ExIR Approaches

9.3 Existing EXIR approaches vs. CFIR

The existing literature aims to explain the significance of a document, a set of documents, or a pair of documents through various explanation methods. Nonetheless, our proposed approach diverges fundamentally from prior work in that we seek to demonstrate how the absence or frequency of certain tokens impacts document relevance. In this section, we examine whether there is any intersection between the two sets of tokens described earlier.

Pointwise Explanation Approach As outlined in Section 2.2, existing pointwise explanation methods elucidate why a specific document aligns with a given query within a retrieval model. Similarly, our proposed approach operates on individual documents and queries, albeit with a distinct objective. Here, we analyze the overlap between the explanations generated by the pointwise explanation method and those derived from our model, as presented in Table 7. This comparison was conducted on 50 pairs of documents.

Listwise Explanation Approach In Section 2, it is explained that listwise explanations typically aim to demonstrate the relevance of a list of documents to a given query. In listwise setup, one set of explanation terms are extracted for a list of documents, a query, and a retrieval model. Conversely, in our approach, we generate distinct explanations for each query-word pair. Therefore, to compare listwise explanations with our method, we aggregate all individual explanations obtained for each document-query pair in the list to create a unified explanation set for the entire list corresponding to a query. The resulting overlap is presented in Table 7.

9.4 Counterfactual Optimization Framework

The different parts of Equation 1 are described here. The $yloss$ in Equation 1 is a hinge loss function as defined in Equation 5. In Equation 5 z is -1 when $y = 0$ otherwise, $z = 1$. $logit(f(c))$ is the logit values obtained from the ML model when the counterfactual c is given as input.

docID	Feature Vector
3686955	[prohibition:2.0, amendment:2.0, under:1.0, dwindled:1.0, eighteenth:1.0, repeal:1.0, repealed:3.0, states:1.0, 1933: 1.0, ratification: 1.0]
6159679	[membrane:5.0, lipids:3.0, remainder:2.0, proteins:3.0, biochemical:2.0, 80:2.0, role:2.0, percent:2.0]
5217641	[waves:6.0, transverse:5.0, electromagnetic:3.0, oscillations:2.0, vibrations:2.0, travel:2.0, radiation:2.0, angles:2.0, transfer:2.0, types:3.0]

Table 6: Sample Feature Vector Corresponding to three different documents

Existing Explanation Methods	Word Overlap
PointWise Explanation (Singh and Anand, 2019)	21.46%
ListWise Explanation (Lyu and Anand, 2023)	9.57%

Table 7: Comparison of CFIR with Existing ExIR Approaches

$$yloss = max(0, 1 - z * logit(f(c))) \quad (5)$$

The distance function ($dist(c_i, x)$) in Equation 1 is computed using the formula given in Equation 6. In Equation 6, d_{cat} represents the number of categorical variables used in the counterfactual input. In Equation 6, the value of I is equal to 1 if the corresponding value of the categorical variable is same in both the counterfactual input c and the original input x, otherwise it is set to 0.

$$dist(c, x) = \sum_{p=1}^{d_{cat}} I(c_p \neq x_p) \quad (6)$$

The diversity in above equation is defined by the formula described in Equation 7. In equation 7, $K_{i,j}$ is equal to $\frac{1}{1+dist(c_i,c_j)}$. $dist(c_i, c_j)$ calculates the distance between two counterfactuals c_i and c_j.

$$div(c_1, \ldots, c_k) = \sum_{i,j} det(K_{i,j}) \quad (7)$$

9.5 Parameters for Counterfactual Setup

The value of λ_1 and λ_2 is set to 1 and 0.5 respectively in Equation 1. The value of k in Equation 1 is set to $k = 3$. In all our experiments in Table 2, we have observed that for $K = 3$ and onward we have always found a counterfactual explanation for each query-document pair where only words were added for the desired counterfactual outcome.

9.6 Adversarial Attacks vs. Counterfactual Explanation

Here we show the performance of our proposed counterfactual explanation approach with an existing adversarial model named PRADA (Wu et al., 2022a). We use the MSMARCO passage dataset as the target corpus. We use same test set (as described in Table 1) as used in the first column of Table 2 in this experiment. Table 8 shows the results in terms of Fidelity score.

Retrieval Model	FD in PRADA	FD in CFIR
ColBERT	74%	75%
MonoT5	80%	82%

Table 8: Performance of CFIR vs. Adversarial Attack Model PRADA (Wu et al., 2022a)

$Imp(d)$ Approach	FD
TFIDF	74%
KeyBERT	70%
BERTSim	**75%**

Table 9: Performance of Different Approaches in $Imp(d)$.

9.7 Implementation of Imp(d)

We explored three ways to compute the top n words from each document. Each one of them is described as follows.

TF-IDF Approach: In this approach we choose top n words from a document based on their TF-IDF weight.

KEYBERT Approach: In this approach we use the model proposed in (Grootendorst, 2020) to extract keywords from a string.

BERT-Based Similarity(BERTSim): In this approach we compute the similarity between the BERT based representation of the query text and each token of the document and then we sort all the tokens based on the similarity.

Table 9 shows the performance of the above-mentioned three approaches in MSMARCO passage dataset and ColBERT retrieval model. $n = 10$ for the experiments shown in Table 9. From Table 9, we can conclude that the BERT-based similarity approach works the best for the $Imp(d)$ function. hence for all the results reported in Table 2, we use the BERTSim approach in the $Imp(d)$ function.