Counterfactual Explanations for Ranking Models

Joel Rorseth
joel.rorseth@uwaterloo.ca
University of Waterloo
Waterloo, Ontario, Canada

ABSTRACT

In recent years, many of the significant advances in artificial intelligence models have come at the cost of increased architectural complexity. Naturally, this complexity has jeopardized the inherent explainability of model decisions, and has inspired a line of work dedicated to the reverse engineering of such decisions, for otherwise uninterpretable models. Though several explainability methods have been adapted to the ranking problem, these solutions produce simple rankings of terms in a given document, scored by their influence on a model's relevance judgement. However, these saliency approaches fail to identify terms vital to maintaining a positive relevance judgement. To this end, we propose the first application of counterfactual explainability to the ranking problem, in order to identify term subsets whose removal from a document results in a non-relevant judgement. We build upon the SEDC counterfactual explanation framework, and adapt novel solutions to accommodate the challenges of indexing, ranking, and efficient representation. We find that our explainability method consistently discovers succinct and pertinent explanations, and in many cases, is able to render them within several seconds. Despite our diverse evaluation, many choices specific to our implementation may influence performance, leaving room for future work to explore alternative indexing tools, ranking models, and datasets.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**; **Content analysis and feature selection**.

KEYWORDS

explainability, counterfactual explanations, document ranking,

ACM Reference Format:
Joel Rorseth. 2022. Counterfactual Explanations for Ranking Models. In *Proceedings of CS848 High Recall Information Retrieval Winter 2022. (CS848 'W22).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

With the rise of deep learning in recent years, the use of deep neural networks has been explored for many aspects of information retrieval (IR). Among other applications, deep learning has been used for ranking (known as *learning to rank*), query modeling, and document representation. These applications have leveraged the increasingly deep architectures proposed in the deep learning literature, which have grown both in computational complexity and architectural complexity. While the increase in computational complexity has been supported and enabled by proportional advances in computer hardware, architectural complexity remains equally complex to humans, especially for non-technical end users. Thus, deep learning research in recent years has become increasingly concerned with the *interpretability* of increasingly complex machine learning models.

More concretely, the study of *explainable artificial intelligence (XAI)* has emerged in recent years, with the goal of improving the interpretability of artificial intelligence models through various types of explanations [5]. The explanations yielded by XAI methods ultimately describe the decision-making process behind an otherwise uninterpretable artificial intelligence model. When sufficiently interpretable to enable understanding in non-technical end users, XAI techniques enable many auditing use cases, such as explaining unexpected decisions or bias exhibited by the model.

Although *explainability* has been popularized in the context of artificial intelligence, many algorithms and software systems that do not employ artificial intelligence may necessitate explanations to enable interpretability. Indeed, *traditional* information retrieval ranking models, such as BM25, are still popular, despite using various statistics and heuristics over artificial intelligence. More importantly, XAI is often motivated by a need to understand *black box* models, in which the nature or characteristics of the model are completely unknown. In this work, we present an explainability method that yields explanations for black box ranking models, supported both traditional and learning to rank alike. For arbitrary ranking models, our method explains why a document was deemed *relevant* to a given query, and justifies its reasoning in terms of document terms.

As opposed to the popular *salience-based* explainability methods, which have been adapted to the ranking problem in previous works [17] [19], our method yields *counterfactual* explanations, which have yet to be explored in the context of ranking. While salience methods yield scores (ie. a ranking) for input features (ie. document terms), counterfactual methods yield a minimal set of features whose removal flips the model prediction (ie. *relevant* to *non-relevant*). To help compute such explanations, we adapt the recently proposed *SEDC* counterfactual explanation algorithm [9] to the ranking problem, accommodating unique aspects of the ranking problem such as relevance, indexing, and re-ranking.

In summary, our main contributions are as follows:

- We develop a counterfactual explanation method for ranking models, the first of its kind, building upon the established SEDC framework.

CS848 'W22, Jan 05–Apr 30, 2022, Waterloo, ON

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Table 1: Examples of Counterfactual Explanations for CORD-19 (TREC-COVID) Queries

Query	Sample Relevant Document Explanation
coronavirus origin	{ origin }
how do people die from the coronavirus	{ like, peopl, die }
what alcohol sanitizer kills coronavirus	{ utmost, investig, transmiss, sanit, alcohol }
coronavirus early symptoms	{ recov, 7, human, tract, data, symptom, earli }
coronavirus hydroxychloroquine	{ hydroxychloroquin }
difference between coronavirus and flu	{ flu }

- We formulate novel strategies to adapt counterfactual methodology to the ranking problem, solving inherent incompatibilities due to indexing, ranking, and representation.
- We perform an extensive evaluation of our method, across a variety of datasets and ranking models, and adapt metrics proposed in other works to the ranking problem.

2 RELATED WORK

A wide variety of XAI research has emerged in recent years, and has inspired efforts to provide explanations for otherwise uninterpretable black box models. Many taxonomies exist to characterize these explanation approaches in different ways. As it concerns the primary subject, *models* (be they AI models or otherwise), explainability solutions can take form in *inherently explainable models* or *post-hoc explainability models*. The former include highly interpretable machine learning models, such as decision trees, while the latter yields explanations by reverse engineering black box (uninterpretable) models. In general, either form can be further categorized into *global* or *local* explanations. Global explanations provide insight into the general behaviour and decision-making process of models, while local explanations describe how a single decision (prediction) was made by the model. In this work, we develop a local post-hoc explainability technique that explains relevance predictions made by black box ranking models.

Perhaps the most significant distinction between local explainability models is the format of the explanation itself. Popular explainability approaches have aligned with *saliency* or *counterfactual* explanation methodologies. Among popular saliency methods are the LIME [16] and SHAP [6] approaches, which aim to determine scores for each feature in the prediction input. By contrast, counterfactual explanations specify how the input must be be modified in order to produce a different prediction [21] [9]. Several works have studied counterfactual explanations more generally [21] [9], as well as having applied them to a variety of problems [18]. Recent counterfactual algorithms SHAP-C and LIME-C [15] have adapted counterfactual methodology to textual and behavioural data, using LIME and SHAP as search heuristics. These works build directly upon the earlier efforts of the SEDC model [9], which bridged counterfactual explanations to the realm of textual data. While useful for IR datasets and corpora, several aspects of these recent methods must be adapted for the ranking problem, which indeed is the novel focus of our proposed solution.

In most works, counterfactual explanation methods expect a subject model in the form of a binary classifier, allowing the explanation algorithm to determine which feature subsets can be removed (or modified) to *flip* the predicted class of a given sample. Often, minimal counterfactual explanations are desired, where minimality is determined by the number of removed features, or the distance between modified and original feature values. In our work, we build upon the SEDC counterfactual explanation approach [9], seeking minimally sized feature subsets that flip the model prediction upon removal. In our adaptation of SEDC to ranking, features are vocabulary terms (ie. words), used to explain ranking model relevance judgements (ie. relevant vs. non-relevant) for query-document pairs.

Among the general XAI approaches aforementioned, several have been adapted to the ranking problem. Recent saliency explanations techniques EXS [17] and LIRME [19] adapt Lime to the explanation of pointwise ranking models. In contrast, our approach adapts SEDC, and produces a counterfactual explanation. While EXS and LIRME produce salience scores for vocabulary terms in a query-document pair, our approach produces a set terms whose removal renders a relevant document irrelevant. To the best of our knowledge, only one other counterfactual effort has been explored in the IR literature. The recently proposed ACCENT framework produces counterfactual explanations for neural recommendation models [18], and is a direct extension of the Prince [4] by several of the original authors. While the recommendation problem is semantically related to ranking, these approaches formulate explanations in terms of actions performed by a user, such as purchasing or reviewing an item in an online shopping recommendation system. Our explanations are formulated in the context of text documents and document ranking, and are strictly subtractive in their search strategy. To the best of our knowledge, our approach is the first counterfactual explanation technique dedicated to explaining query-document ranking models.

3 COUNTERFACTUAL EXPLANATION FOR RANKING

In this work, we focus on the problem of explaining document rankings. More specifically, given a corpus of documents and queries, we aim to generate explanations as to why a document was deemed relevant to a query, by a given ranking model. Rather than justifying the ranking with respect to the query terms, we specifically justify the relevance of a selected document with respect to the terms in the document.

Applying counterfactual methodology to this problem, we further refine the formulation to specify the format of the explanation. For a given document deemed relevant to a given query (by a given

model), produce a subset of document terms whose removal renders the document non-relevant. Furthermore, since the number of possible solutions is exponential in the number of unique terms, produce a fixed set of such solutions while seeking those with minimal size (ie. the fewest number of terms removed).

In Table 1, we list a sample of queries from the COVID-19 Open Research Dataset (TREC-COVID) [20, 22], alongside the counterfactual explanation generated by our algorithm for their highest ranked document. From these explanations, many interesting trends can be inferred about the ranker and corpus documents. For example, the top ranked document for query "how do people die from the coronavirus" is relevant due to the presence of key terms "peopl" and "die", though terms common to all documents, such as "coronavirus", are not found to be salient. In the query "what alcohol sanitizer kills coronavirus", many terms are implicated in the explanation in addition to those derived from the query. This suggests that the ranker was particularly lenient in its assessment of relevance, selecting documents that contained relatively few key terms.

4 METHOD

In the following subsections, we describe the components of the proposed ranking explanation system. At a high level, an index is created to provide access to a selected corpus of documents and queries. A predefined ranking model is instantiated, which is employed to generate initial top-k document rankings for all queries in the index. Document-query pairs are drawn from the index, and are ultimately passed to the SEDC algorithm to be explained. To generate counterfactual explanations, the SEDC algorithm invokes the ranker repeatedly using an in-memory re-ranker, which re-ranks the top-(k+1) documents, each time substituting the document being explained with a perturbed copy.

4.1 Indexing

In the first step of our explanation pipeline, an index is built to facilitate efficient storage and retrieval of documents and queries, among other dataset artifacts. To build and interact with this index, we employ the PyTerrier library [8], a Python API for the popular Terrier retrieval platform [12]. The datasets used in our experiments are obtained using the ir_datasets library [7], which provides an interface to load a variety of popular benchmark datasets into PyTerrier.

As it concerns our framework, the Terrier indexing process constructs a list of documents D and a list of queries Q. Documents and queries are made available for random access via the PyTerrier API, whom achieves this by assigning documents and queries unique identifiers $d \in [0, |D| - 1]$ and $q \in [0, |Q| - 1]$, respectively. Therefore we denote the document with id i as $D[i]$, and the query with id j as $Q[j]$. Our work adopts a bag-of-words representation for documents and queries, thus each element of D and Q is an unordered list containing all unique terms present.

Through the PyTerrier API, a list of document terms (vocabulary) is made available for random access as well. Each term in the vocabulary V is stored at a unique index $v \in [0, |V| - 1]$, therefore $V[k]$ denotes the term with implicit identifier k. We remove morphological and inflexional endings (ie. suffixes) from words, by allowing Terrier to apply an implementation of the Porter Stemming algorithm [14] during the initial indexing process.

In addition to the indexed data made available by PyTerrier, we extract and process certain data required for efficient manipulation by SEDC. The document and vocabulary lists D and V enable the creation of a sparse document corpus, a $|D|x|V|$ matrix M denoting the bag-of-words representation for $|D|$ documents. This matrix is composed of 0 and 1 values, as defined in Equation 1, which denote the presence or absence of a term in a given document, respectively. This matrix facilitates random access extraction of sparse document representations, which are essential to the tractable exploration of large corpora.

$$M[i,k] = \begin{cases} 1 & V[k] \in D[i] \\ 0 & otherwise \end{cases} \tag{1}$$

4.2 Ranking

In this work, we consider ranking models that output top-k rankings. While previous saliency-based explanation approaches for ranking have focused on pointwise learning-to-rank rankers [17] [19], our solution is deliberately agnostic to the underlying model architecture. Our method supports any ranking model that produces a list of top-k documents D_q^k for a given query q. However, the SEDC counterfactual explanation algorithm will re-rank repeatedly, in order to find counterfactual explanations for a given document. Therefore, the chosen ranker should support efficient re-ranking capable of circumventing index recompilation.

In keeping with their typical definition, counterfactual methods such as SEDC explain the predictions of *binary* classifiers. In a document ranking problem, documents are assigned numeric scores (ranks) with respect to a given query, thus in order to treat the ranking model as a binary classifier, each score must be mapped to a binary variable. Naturally, we impose a transformation to discretize scores into relevance assessments (ie. relevant or non-relevant). We utilize the *Top-k Binary* relevance transformation suggested in recent works for saliency-based ranking explanations [17].

Let $R(q, d)$ be a function that yields the ranking model score for a given document d, with respect to D_q^k. Let X be a random variable representing the *relevant* or *non-relevant* outcome for d. As defined in Equation 2, a document is deemed relevant only if the model ranks the document among the top-k documents, for some predefined query and k value, and is deemed non-relevant otherwise.

$$P(X = relevant|q, d, R) = \begin{cases} 1 & R(q,d) \geq R(q,d_k) \\ 0 & otherwise \end{cases} \tag{2}$$

In our experiments, we employ PyTerrier once again for ranking. Terrier supports an extensive variety of traditional weighted ranking models out of the box, which can be used to generate a baseline ranking, and in turn, an in-memory re-ranked ranking, The Terrier team has made available several extensions that implement various deep learning (DL) learning-to-rank models, which may also be integrated in the re-ranking stage, though we leave this integration for future work.

4.3 SEDC

Until recently, counterfactual explanation methods were generally incapable of interpreting textual data, such as document corpora. In such cases, a unique feature is required for each term in the corpus vocabulary. Previous counterfactual approaches have been enabled by considering only a small, finite set of *binary* features, however the vocabulary (feature) space necessary for textual data is orders of magnitude larger. The SEDC approach was the first to solve this issue, and therefore facilitate explanation of textual data. SEDC utilizes sparse document representations, which efficiently denote vocabulary terms present in a given document, and thereby support tractable explanation (in particular, tractable exploration of the feature powerset). Intuitively, this approach utilizes a *bag-of-words* representation, by means of sparse matrices and vectors.

To generate a local explanation for a given document-query pair, SEDC must navigate a large search space, defined as the powerset of words contained in all documents. To explain a given document, SEDC reduces this initial set of words to those contained in the document itself. As a feature subset of minimal size is preferred, the general algorithmic approach involves iterating through subsets of increasing size, beginning with all feature subsets of size 1. For each subset, all terms in the subset are removed from the document being explained, which is subsequently ranked again to test for non-rel judgement. This secondary ranking process is described in more detail the following subsection. This algorithm guarantees that the first non-rel judgement constitutes a minimal counterfactual explanation, though its complexity is exponential in the number of document terms. SEDC incorporates several heuristics to reduce this complexity, namely to prune the search space, and to guide the search via local improvement calculation.

4.4 Generating Explanations

Before generating explanations, several configuration settings are established, namely the value of k (ie. top-k), the dataset, and the ranking model. The selected dataset is indexed using PyTerrier, and as described, is used to create the sparse document corpus. The ranking model is invoked to generate a base ranking D^{k+1}, denoting the top-(k+1) documents for each query $q \in Q$, D_q^{k+1}. Our SEDC explanation algorithm proceeds with the generation of explanations, one query-document pair at a time, making sure to utilize the sparse representations.

For each pair, SEDC iterates the powerset of document terms as described previously, and assesses the relevance of each modified document copy (ie. subset of document terms). The efficient determination of relevance is achieved by recalling D_q^{k+1} for the current query q. Since the original document being explained, d, was deemed relevant, $d \in D_q^{k+1}$. We allow SEDC to modify d as described, removing terms from its sparse bag-of-words representation. The modified document, d', is substituted in place of d, and the resulting set of k+1 documents is re-ranked using PyTerrier's in-memory TextScorer re-ranker. The relevance of d' is calculated in the context of the new ranking $D_q'^{k+1}$, as specified in Equation 3. The modified document d' is no longer relevant if it no longer ranks among the top-k documents (as d did).

Table 2: Experiment Datasets

Dataset	# Documents	# Queries
NFCorpus (dev)	5.4K	325
CORD-19 (TREC-COVID)	193K	50
WikIR (en1k test)	370K	100
MSMARCO Document (dev)	3.2M	5.2K

$$P(X = relevant|q, d', R) = \begin{cases} 1 & R(q, d') \geq R(q, d_k) \\ 0 & otherwise \end{cases} \quad (3)$$

After finding one or more modified documents with minimal modification, SEDC outputs all found explanations, along with their computation time and other statistics. Finally, the size of the smallest explanation is also reported, known formally as the switching point [11], for use in our experiments.

5 EXPERIMENTS

In this section, we describe the experiments performed, including the metrics for evaluating counterfactual explanations, datasets used, and trends observed in the results. Since, to the best our knowledge, our approach is the first counterfactual explanation method for the ranking problem, our experiments are unable to compare trends and performance with other methods. In lieu, our experiment profiles our method and the explanations generated, by adapting metrics established in the counterfactual explainability literature. We execute all experiments on a Linux server running Ubuntu 18.04.5 LTS, with 2x Tesla P40 GPUs, then write and run all code using Python 3.6.9.

5.1 Metrics

To benchmark our explanation method, we considered several metrics proposed in the literature, across various applications of counterfactual explainability which have yet to standardize any evaluation metrics. In addition, many metrics proposed are specific to certain types of data, such as numeric or categorical. Not all counterfactual applications limit their exploration of perturbed instances to subtractive perturbation, thus several proposed metrics attempt to capture the magnitude of feature modifications, which simply do not occur in other approaches.

Several metrics are proposed in the paper introducing the DiCE explanation tool [10]. The proposed *validity* metric measures the fraction of explanations which are in fact counterfactuals, however all explanations returned by our method are valid counterfactuals. *Sparsity* and *proximity* metrics are also proposed, which aim to capture the number of features that have changed, and by how much, respectively.

Since our method only removes, but does not modify features (document terms), a straightforward adaptation of the proximity metric is not inherent for our adaptation to textual data. However, the sparsity metric can be directly observed via the number of removed features. An equivalent metric known as the *switching point* has been established and utilized in other works [11] [15], and will serve as the primary indicator for explanation quality in

our experiment. This metric directly rewards explanations for removing fewer features, which not only results in more concise (and therefore interpretable) explanations, but also penalizes trivial explanations that are prohibitively destructive (eg. an explanation that removes *all* features would render a document non-relevant, but this is no revelation). Additionally, we measure the performance of our method through the time required to generate each explanation.

5.2 Rankers and Datasets

To gauge performance, we run experiments with several ranking models and datasets. In particular, we use the BM25 and PL2 traditional ranking models, along with four datasets of various size summarized in Table 2. The BM25 and PL2 weighting models incorporate different statistical calculations, and may therefore exhibit different behaviours and tendencies that can be explained. The variety in dataset sizes helps to identify performance limitations and boundaries for our method, as well as how performance changes with respect to top-k value.

When measuring switching point, we evaluate our method for BM25 and PL2 rankers on the CORD-19 (TREC-COVID) dataset [20, 22], as well as the WikIR (en1k test) dataset [2, 3]. To measure computation time, we also evaluate our method against the NFCorpus (dev) [1] and MSMARCO Document (dev) [13] datasets. In the experiments, we select a predetermined sample of 50 queries set aside in each dataset, guaranteeing that all tests on a given dataset evaluate the same query-document pairs. Furthermore, we select the top-ranked document for each of the 50 queries, resulting in 50 total explanations for each experiment. Each experiment is parameterized by a value of k (ie. top-k), a dataset, and a ranker.

5.3 Results

5.3.1 Switching Point. As it concerns switching point, our method maintains an average explanation size between 1 and 2 terms, for all tested values of k. These results are summarized in Table 3, which illustrates a clear linear correlation between the value of k and the mean switching point. This trend is intuitive, since it becomes more difficult for a document to be deemed non-relevant as k increases towards $|D|$, at which point all documents are deemed relevant. Therefore, we empirically find that, as k increases, more salient terms must be removed in order to solicit a non-relevant assessment by the ranker.

Although not denoted in Table 3, all switching point median values remain at exactly 1. This indicates that, across all tested top-k values, the majority of explanations have the minimum possible size of 1 term. In other words, several outlier explanations inflate the average towards 2 terms in tests with larger top-k values, however the majority of explanations remain unchanged with respect to k. This is supported by another clear trend, in which the standard deviation generally increases as k increases. In both datasets, the increasing standard deviation reflects the growing presence of outliers, though rarely to extremes past 2 terms.

5.3.2 Computation Time. Evaluating the performance of our explanation method, we observe significant variance in computation time depending on the dataset and value of k. This variance is clear across our experiment results, listed in Table 4. For MSMARCO, our largest dataset, computation only completed in the top-1 experiment, while the CORD-19 experiments experienced no such issues. Other larger experiments did not complete as well, and are denoted as "N/A" in Table 4. In these cases, the Terrier platform simply crashed and was unable to continue, due to resource constraints.

Table 3: Switching Point Results

Dataset	Top-k	Ranker	Mean	Std. Dev.
CORD-19 (TREC-COVID)	top-1	BM25	1.000	0.000
		PL2	1.000	0.000
	top-10	BM25	1.000	0.000
		PL2	1.300	1.269
	top-50	BM25	1.400	1.497
		PL2	1.563	1.306
	top-100	BM25	1.633	1.746
		PL2	1.957	1.856
WikIR (en1k test)	top-1	BM25	1.000	0.000
		PL2	1.000	0.000
	top-10	BM25	1.082	0.566
		PL2	1.082	0.566
	top-50	BM25	1.553	1.699
		PL2	1.979	2.274

While the number of documents in the MSMARCO corpus greater than the others by orders of magnitude, as reported in Table 2, we note that the number of documents $|D|$ is not the sole culprit of increased computation time. NFCorpus contains orders of magnitude less documents than the other datasets, yet fails to handle top-50 or top-100 experiments. Intuitively, the impact of the number of documents and queries can be ruled out, due to the fact that SEDC explains only a single query-document pair at a time.

We hypothesize that the distribution of corpus document sizes is the predominant static factor behind computation time, which would be multiplied by the computational efficiency of the ranking model itself. Given the wide variety of ranking models in the literature, performance could vary wildly, just as they would in real-world search engines. In our implementation, the re-ranking process performed repeatedly by PyTerrier (at the request of SEDC) may be less efficient than other ranking models whom choose to (re-rank) in different ways. In spite of these factors, top-1 and top-10 experiments yield explanations for a query-document pair in a matter of seconds, in as few as 2 seconds, or as many as 83.

In Figure 1, the distribution of computation time with respect to k is plotted, highlighting several trends. With larger values of k, the median computation time increases, as expected, however the variance also increases proportionally. Outlier explanations with wildly varying computation time become more common, meaning that many documents retain small explanations, and many necessitate larger. In general, the computation time appears to increase superlinearly with increasing k, which reflects the ability of the ranking model to scale with k.

6 CONCLUSION

In this paper, we set out to create the first model-agnostic local counterfactual explanation method for ranking model assessments.

Table 4: Average Computation Time

Top-k	CORD-19	WikIR	NFCorpus	MSMARCO
top-1	2.011	2.484	9.701	32.388
top-10	3.639	9.197	51.366	N/A
top-50	25.168	68.778	N/A	N/A
top-100	82.693	N/A	N/A	N/A

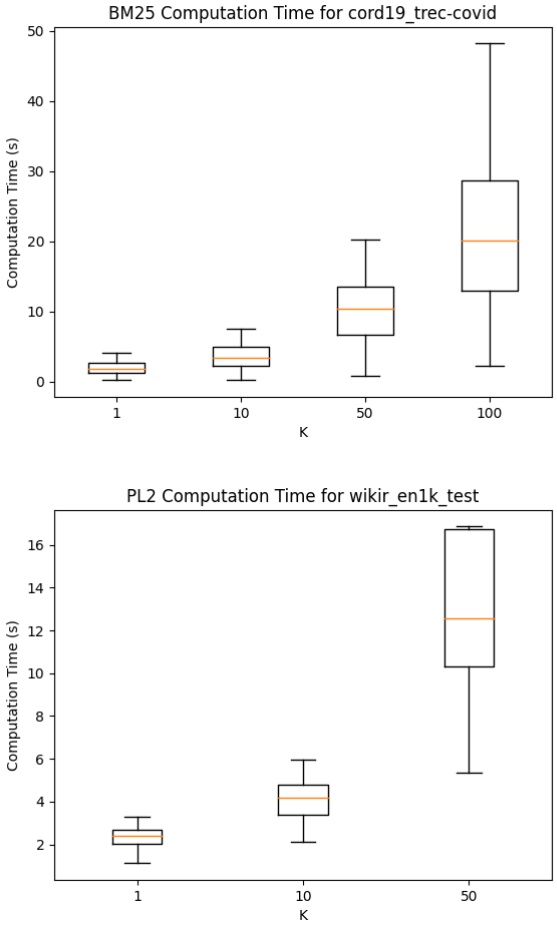

Figure 1: Computation Time for Increasing Top-K Values.

Although several explainability methods have been proposed in the XAI literature, few standards exist within this relatively new area of study. Fewer yet are those that incorporate a counterfactual approach, as opposed to the popular saliency-based approach, or those capable of handling textual data. These challenges are the primary motivation of this paper, though are equally matched by the difficulties of adapting counterfactual explanations to the ranking problem. We aim to adapt existing counterfactual algorithms for ranking models, as opposed to the generic AI classifiers assumed in most XAI works, as well as to adapt metrics and formulations from the literature.

In specific, we present a novel adaptation of the SEDC counterfactual explanation algorithm for ranking models. To enable this unique application of XAI, we build a framework to adapt the ranking problem to binary classification, in order to interface with a tool that is primarily intended for simpler AI classifiers. We present solutions for additional complexities unique to the ranking problem, such as indexing and the efficient representation of textual data. In addition to this formulation, several experiments are performed to evaluate our method, using metrics adapted from other applications of counterfactual XAI.

We find that, across several datasets of vastly different size and scale, the vast majority of counterfactual explanations produced by our method maintain a minimal size of 1 document term, guaranteeing concise and meaningful insights for non-technical users. The computation time of our method does vary greatly for different top-k values, and appears to increase superlinearly with increasing k. Despite this growth, explanations are generated within several seconds in many cases, especially for top-1 or top-10 experiments.

By virtue of being the first adaptation for the ranking problem, we identify many challenges for the generation of counterfactual ranking explanation in future work. Although explanations maintain minimal size in most of our experiments, more work is required to understand and reduce the computational time, in order to enable use by potentially impatient non-technical users. In specific, other indexing platforms, ranking model frameworks, and ranking model architectures should be explored, and could easily be substituted within the broader framework we have created. Many other promising counterfactual methods from the XAI literature claim to offer improvements over SEDC, such as the recent SHAP-C and LIME-C textual counterfactual algorithms [15].

In addition to other promising algorithms, we believe that metrics from other counterfactual works could be reformulated for textual data, or for the ranking problem more specifically. For example, the proximity metric [10] measures how significantly each (numerical or categorical) explanation feature was modified, however a similar distance could be derived for textual data using edit distance. Indeed, incorporating proximity necessitates a counterfactual approach that permits feature modification, rather than removal alone. This alternative could make for interesting future work, however it is unclear term modification is a reasonable explanation in a retrieval context.

REFERENCES

[1] Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A Full-Text Learning to Rank Dataset for Medical Information Retrieval. In *Proceedings of the European Conference on Information Retrieval (ECIR)* (Padova, Italy). Springer.
[2] Jibril Frej, Didier Schwab, and Jean-Pierre Chevallet. 2020. MLWIKIR: A Python Toolkit for Building Large-scale Wikipedia-based Information Retrieval Datasets in Chinese, English, French, Italian, Japanese, Spanish and More. In *CIRCLE*.
[3] Jibril Frej, Didier Schwab, and Jean-Pierre Chevallet. 2020. WIKIR: A Python toolkit for building a large-scale Wikipedia-based English Information Retrieval Dataset. In *LREC*.
[4] Azin Ghazimatin, Oana Balalau, Rishiraj Saha Roy, and Gerhard Weikum. 2020. Prince: Provider-side interpretability with counterfactual explanations in recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 196–204.
[5] David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. 2019. XAI—Explainable artificial intelligence. *Science Robotics* 4, 37 (2019), eaay7120.
[6] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems* 30 (2017).

[7] Sean MacAvaney, Andrew Yates, Sergey Feldman, Doug Downey, Arman Cohan, and Nazli Goharian. 2021. Simplified Data Wrangling with ir_datasets. *InSIGIR*.
[8] Craig Macdonald and Nicola Tonellotto. 2020. Declarative Experimentation inInformation Retrieval using PyTerrier. In *Proceedings of ICTIR 2020*.
[9] David Martens and Foster Provost. 2014. Explaining data-driven document classifications. *MIS quarterly* 38, 1 (2014), 73–100.
[10] 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*. 607–617.
[11] Dong Nguyen. 2018. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 1069–1078.
[12] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Douglas Johnson. 2005. Terrier information retrieval platform. In *European Conference on Information Retrieval*. Springer, 517–519.
[13] Nick Craswell Li Deng Jianfeng Gao Xiaodong Liu Rangan Majumder Andrew McNamara Bhaskar Mitra Tri Nguyen Mir Rosenberg Xia Song Alina Stoica Saurabh Tiwary Tong Wang Payal Bajaj, Daniel Campos. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *InCoCo@NIPS*.
[14] Martin F Porter. 1980. An algorithm for suffix stripping. *Program* (1980).
[15] Yanou Ramon, David Martens, Foster Provost, and Theodoros Evgeniou. 2020. A comparison of instance-level counterfactual explanation algorithms for behavioral and textual data: SEDC, LIME-C and SHAP-C. *Advances in Data Analysis and Classification* 14, 4 (2020), 801–819.
[16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.
[17] 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*. 770–773.
[18] Khanh Hiep Tran, Azin Ghazimatin, and Rishiraj Saha Roy. 2021. Counterfactual Explanations for Neural Recommenders. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1627–1631.
[19] Manisha Verma and Debasis Ganguly. 2019. LIRME: locally interpretable ranking model explanation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1281–1284.
[20] E. Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, W. Hersh, Kyle Lo, Kirk Roberts, I. Soboroff, and Lucy Lu Wang. 2020. TREC-COVID: Constructing a Pandemic Information Retrieval Test Collection. *ArXiv* abs/2005.04474 (2020).
[21] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.* 31 (2017), 841.
[22] Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, K. Funk, Rodney Michael Kinney, Ziyang Liu, W. Merrill, P. Mooney, D. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, B. Stilson, A. Wade, K. Wang, Christopher Wilhelm, Boya Xie, D. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. CORD-19: The Covid-19 Open Research Dataset. *ArXiv* (2020).

,