PRADA: Practical Black-Box Adversarial Attacks against Neural Ranking Models

CHEN WU, RUQING ZHANG, and JIAFENG GUO*, Institute of Computing Technology, Chinese Academy of Sciences; University of Chinese Academy of Sciences, China
MAARTEN DE RIJKE, University of Amsterdam, The Netherlands
YIXING FAN, and XUEQI CHENG, Institute of Computing Technology, Chinese Academy of Sciences; University of Chinese Academy of Sciences, China

Neural ranking models (NRMs) have shown remarkable success in recent years, especially with pre-trained language models. However, deep neural models are notorious for their vulnerability to adversarial examples. Adversarial attacks may become a new type of web spamming technique given our increased reliance on neural information retrieval models. Therefore, it is important to study potential adversarial attacks to identify vulnerabilities of NRMs before they are deployed.

In this paper, we introduce the Word Substitution Ranking Attack (WSRA) task against NRMs, which aims to promote a target document in rankings by adding adversarial perturbations to its text. We focus on the decision-based black-box attack setting, where the attackers cannot directly get access to the model information, but can only query the target model to obtain the rank positions of the partial retrieved list. This attack setting is realistic in real-world search engines. We propose a novel Pseudo Relevance-based ADversarial ranking Attack method (PRADA) that learns a surrogate model based on Pseudo Relevance Feedback (PRF) to generate gradients for finding the adversarial perturbations.

Experiments on two web search benchmark datasets show that PRADA can outperform existing attack strategies and successfully fool the NRM with small indiscernible perturbations of text.

CCS Concepts: • **Information systems** → **Retrieval models and ranking**; **Adversarial retrieval**.

Additional Key Words and Phrases: Adversarial attack, Decision-based black-box attack setting, Neural ranking models

ACM Reference Format:
Chen Wu, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2022. PRADA: Practical Black-Box Adversarial Attacks against Neural Ranking Models. *ACM Transactions on Information Systems* 1, 1, Article 1 (May 2022), 24 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Ranking models are central to information retrieval (IR) research. With the advance of deep neural networks, we are witnessing a rapid growth in neural ranking models (NRMs) [9, 18, 36, 40], achieving new state-of-the-art results in learning query-document relevance patterns. Recent

*Jiafeng Guo is the corresponding author.

Authors' addresses: Chen Wu, Ruqing Zhang, and Jiafeng Guo, {wuchen17z,zhangruqing,guojiafeng}@ict.ac.cn, Institute of Computing Technology, Chinese Academy of Sciences; University of Chinese Academy of Sciences, NO. 6 Kexueyuan South Road, Haidian District, Beijing, China, 100190; Maarten de Rijke, m.derijke@uva.nl, University of Amsterdam, Amsterdam, The Netherlands; Yixing Fan, and Xueqi Cheng, {fanyixing,cxq}@ict.ac.cn, Institute of Computing Technology, Chinese Academy of Sciences; University of Chinese Academy of Sciences, NO. 6 Kexueyuan South Road, Haidian District, Beijing, China, 100190.

1046-8188/2022/5-ART1
https://doi.org/10.1145/nnnnnnn.nnnnnnn

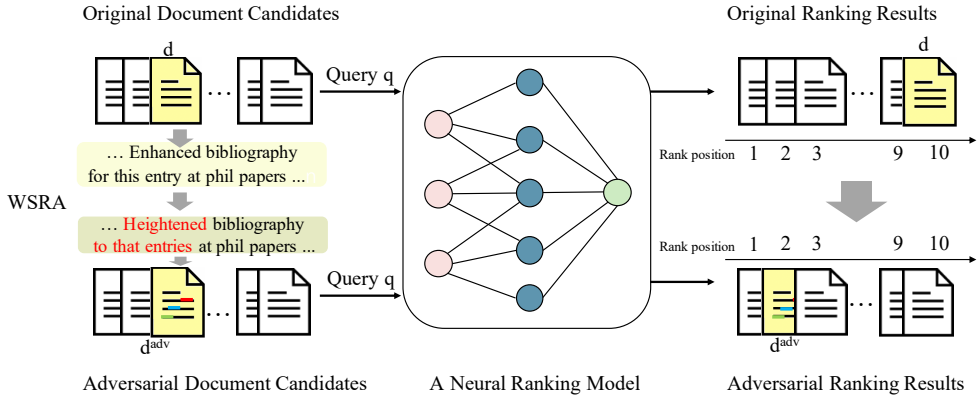

Fig. 1. Demonstration of the WSRA task. Given a neural ranking model, adversarial perturbation is added to the target document d and the adversarial example d^{adv} will be promoted in rankings with respect to the query q.

research has explored pre-trained language models (e.g., BERT [11] and ELMo [44]) in the context of document ranking, and shown that they can achieve remarkable success on a variety of search tasks [17, 24, 32]. The impact of pre-trained models is not limited to academic research. In industry, BERT and, more generally, transformers are being put to practical usage [see, e.g., 29].

Adversarial examples. Deep neural models are notorious for their vulnerabilities to adversarial examples [14, 52]. For example, Goodfellow et al. [14] show that a panda image, added with imperceptible perturbations, is misclassified as a gibbon by GoogLeNet [51]. Liang et al. [28] prove that even tiny modifications to a character or a word can fool state-of-the-art deep text classifiers. Recent observations have also shown that rankings can rapidly change due to small, almost indiscernible changes of documents [15]. Hence, adversarial attacks may become a new type of web spamming [19] in the neural network based methods which gain importance in IR. Since adversarial examples are maliciously crafted by adding perturbations that are imperceptible to humans to legitimate examples, they may not be detected by traditional anti-spamming methods [50]. Up to now, little attention has been paid to adversarial attacks against NRMs, except for analyses of the robustness of ranking models carried out by Goren et al. [15]. Therefore, we believe it is critical to study potential adversarial attacks to identify the vulnerability of NRMs before they are deployed and help facilitate the development of the corresponding countermeasures.

A new adversarial attack task. In this paper, we introduce the Word Substitution Ranking Attack (WSRA) task against NRMs. As shown in Figure 1, given a neural ranking model, the WSRA task aims to promote a target document in rankings with respect to the query by replacing important words in its text with their synonyms in a semantic-preserving way. An effective adversarial sample in WSRA needs to satisfy the following qualities: (1) imperceptible to human judges yet misleading to NRMs; and (2) fluent in grammar and semantically consistent with the original document. We clarify the reason why we focus on word substitutions in this work. We also discuss the major differences between the WSRA task against NRMs and adversarial attacks for image retrieval and text classification. Besides, we define different adversarial settings for the WSRA task in terms of the information that attackers rely on, including *white-box* attacks and *black-box* attacks. The black-box attacks are further divided into *score-based* attacks and *decision-based* attacks. For the evaluation of the WSRA task, we define the Success Rate (SR) metric for the attacking and adapt the Perturbation Percentage (PP) and Semantic Similarity (SS) from Natural Language Processing (NLP) for automatic evaluation.

In this work, we focus on the *decision-based black-box attack setting* for the WSRA task. This attack scenario is realistic and important, because most of the real-world search engines are black-boxes and only output hard rank positions. It is also challenging since the gradient cannot be directly computed and the predicted probability is not provided.

An adversarial ranking attack method. We make the first attempt to address the WSRA task under the decision-based black-box attack setting. Specifically, we introduce a novel Pseudo-Relevance based ADversarial ranking Attack method, or PRADA for short, to generate adversarial samples. The key idea is to learn a surrogate model to imitate the behaviors of the target NRM for finding the adversarial perturbations. Inspired by the Pseudo Relevance Feedback idea [PRF, 10] in IR, we query the target NRM and take the top-ranked results as relevant examples to learn a surrogate ranking model. Then, we identify the important words in a document which have a high influence on the final ranking result via the prior-guided gradients generated by the surrogate model. With the important words, we apply Projected Gradient Descent [PGD, 33] to generate gradient-based adversarial perturbations to the embedding space according to the expected ranking order. Finally, we replace the important word with its synonyms in a semantics-preserving way and repeat this process by iterating the importance words list to find the final adversarial sample.

Experiments. We conduct experiments on two web search benchmark datasets, the MS MARCO document ranking dataset and the MS MARCO passage ranking dataset. We compare with several state-of-the-art adversarial attack strategies and our experimental results show that PRADA can successfully promote the target document in rankings with the highest attack success. At the same time, the perturbation percentage is considerably lower than for competing attack methods while the semantic similarity score is comparably high.

Main contributions. The main contributions of this paper are (1) We introduce a new WSRA task against NRMs for identifying the vulnerability of NRMs and consequently contributing to the design of robust NRMs; (2) We make the first attempt to address the WSRA task under the decision-based black-box attack setting, and propose a novel PRADA method based on PRF to generate adversarial examples; and (3) We conduct rigorous experiments to demonstrate the effectiveness of our proposed model.

2 RELATED WORK

In this section, we briefly review three lines of related work, including web spamming, text ranking models and adversarial attacks.

2.1 Web Spamming

For many queries in the Web retrieval setting, there exists an on-going ranking competition: authors of some Web pages may manipulate their documents so as to have them ranked high [15]. Web spamming refers to these actions of manipulating web pages intended to mislead search engines into ranking some pages higher than they deserve [19]. The consequences of web spamming are that the quality of search results decreases and search engine indexes are inflated with useless pages, which increases the cost of each processed query.

Existing spamming techniques can be divided into *term spamming* and *link spamming* [19]. Term spamming refers to techniques that tailor the contents of a web page's text field (e.g., document body, title, meta tags), in order to make spam pages relevant to some queries [4]. For example, Gyongyi and Garcia-Molina [19] summarized a list of different types of term spamming in the Web, including repetition, weaving, dumping and stitching. Link spamming [53] creates link structures that are meant to increase the importance of one or more of their pages. For example, Link farms [**?**], honey pots [19] and spam link exchange are typical link spamming techniques. To combat such manipulation, prior works studied the detection of web spamming from the perspective of content

analysis [39, 45] and link analysis [3, 20, 54], respectively. For example, different content-based features have been explored to build spam classifiers to detect term spammings [39]. Meanwhile, a variety of trust and distrust propagation algorithms such as Trustrank [**?**] and BadRank [**?**] were proposed to fight against link spamming.

Note that the *spam* is a concept which is different from the *web spam*. While spam refers to unsolicited or undesired electronic messages (e.g., email spam or message), web spam is an IR concept [19] which refers to the web pages that have been manipulated to be ranked higher in search engines. So the works [**? ? ?**] which talk about the spam detection are in fact irrelevant to our work. For example, **?**] leverages the Generative Adversarial Network (GAN) to detect deceptive reviews (e.g., to classify whether a review is a deceptive review). **?**] combined the CNN and LSTM to do spam detection. **?**] propose an ensemble approach which combines global and local features of e-mails together to detect spam effectively. Since they all study about the spam detection, they are different from our work (e.g., as an adversarial attack). So they cannot become our baselines.

The proposed WSRA task can be viewed as a new type of web spamming against NRMs. The difference between the proposed WSRA and traditional web spamming is that our WSRA task promotes a target document in rankings by adding an imperceptible perturbation to its text. As a result, it may not be detected by traditional anti-spamming methods [50].

2.2 Text Ranking Models

Ranking models lie at the heart of research on IR. During the past decades, different techniques have been proposed for constructing ranking models, from traditional heuristic methods [49], probabilistic methods [46, 48], to modern learning to rank methods [25, 30]. For traditional heuristic methods, Query likelihood (QL) [46] model and BM25 [48] are classical ranking models. For example, QL is often based on Dirichlet smoothing [**?**] to model the likelihood of a document being relevant to a given query. For modern learning to rank methods, RankSVM [**?**] and LambdaMART [**?**] are representative pairwise and listwise learning to rank models, respectively. RankSVM is based on Structural Support Vector Machine to solve the ranking problem. And LambdaMART leverages gradient boosting to produce an ensemble of retrieval models.

With the advance of deep learning technology, we have witnessed a substantial growth of interest in NRMs [9, 18, 36, 40], which have shown to provide promising effectiveness improvements compared to previous IR methods. Based on the different assumptions about the feature representation and interaction, existing NRMs can be divided into representation-focused NRMs and interaction-focused NRMs. For example, DSSM [**?**] is a representative-focused deep matching model designed for Web search, which contains a letter n-gram based word hashing layer, two non-linear hidden layers and an output layer. DRMM [18] is an interaction-focused deep matching model designed for ad-hoc retrieval. It consists of a matching histogram mapping, a feed forward matching network and a term gating network. Conv-KNRM [**?**] is an interaction-focused deep matching model, which models n-gram soft matches for ad-hoc retrieval based on convolutional neural networks (CNN) and kernel-pooling. Duet [36] is a hybrid deep matching model which combines both the representation-focused architecture and the interaction-focused architecture.

Recently, pre-trained language representation models such as BERT [11] have been widely adopted for text ranking, showing great success when fine-tuned on a wide range of search tasks [17, 24, 32]. BERT applies the multi-layer bidirectional Transformer encoder architecture for language modeling. For example, monoBERT [**?**] concatenates the query and the document with special token (e.g., [CLS] and [SEP]) as the input to BERT. The relevance score of the document to a given query is then computed by a sigmoid function over the [CLS] representation. Furthermore, ColBERT [**?**] leverages contextualized late interaction over BERT for efficient retrieval. The queries and documents are independently encoded into fine-grained representations that interact via cheap and pruning-friendly computations. In this paper, we use BERT and several representative

NRMs (e.g., Conv-KNRM [**?**] and Duet [36]) as the target ranking models to evaluate the attack effectiveness.

Since neural networks become ever more sophisticated, it is costly to obtain massive amounts of annotated training data. Dehghani et al. [10] proposed to address this problem by taking advantage of existing unsupervised methods such as BM25 [48] for constructing a weakly annotated training set. A neural ranking model was then trained with these weakly annotated training data. Besides, Izsak et al. [21] also studied how can a search engine with a relatively weak relevance ranking function compete with a search engine with a much stronger relevance ranking function. Inspired by the idea, we propose to train the surrogate ranking model with weak supervision signals generated by the target model.

2.3 Adversarial Attacks

Adversarial attacks aim to find a minimal perturbation that maximizes the model's risk of making wrong predictions. Depending on the degree of access to the target model, adversarial examples can be crafted in two different settings: white-box and black-box settings [**?**]. In the white-box attack setting, attackers have complete access to the target model. Early researchers [14, 33, 52] have extensively studied adversarial attacks for continuous data, e.g., images. For example, the Fast Gradient Sign Method [FGSM, 14] utilized the error function of the model output and the target category to generate the adversarial perturbation. Moreover, Projected Gradient Descent [PGD, 33] is an iterative version of FGSM, which is regarded as one of the most powerful attacks [2].

In the black-box attack setting, attackers only have access to the outputs of the target model [6]. Prior work has explored the black-box attack for many NLP tasks, including text classification [13, 28], sentiment analysis [1, 26, 28], and natural language inference [1, 35]. Adversarial attacks for text are challenging due to the discrete input space. To alleviate the problem, Goodfellow et al. [14] adopted FGSM to generate perturbations in the word embedding space and utilized nearest neighbor search to find the closest words. However, such methods treat all words as equally vulnerable and replace them with their nearest neighbors, which leads to non-sensical and word-salad outputs [56]. To tackle the problem, a number of publications [22, 23, 28] have adopted heuristic rules to find important words and substitute these words with synonyms. Note that our work is different from BERT-ATTACK [27]. While BERT-ATTACK conducts the adversarial attack for text *classification*, we conduct the adversarial attack for text *ranking*. Specifically, BERT-ATTACK first finds vulnerable words by the output score difference between the original sentence and the modified sentence (e.g., by masking the important word). Then, it replaces these important words by a test and try method. For example, it iterates over the candidate word list for each important word and checks the output score with the replaced word. The replacement will be kept if it lowers the prediction score. However, this method relies on score-based assumption where attackers can obtain the output score of classification model, which is not practical in ranking. In this paper, we propose to train a surrogate ranking model to obtain the gradient information to conduct the adversarial attack.

Adversarial attacks have also been extensively studied in the context of recommendation systems. Initially, studies [**?**] focused on hand-engineered fake user profiles against rating-based collaborative filtering to harvest recommendation outcomes toward an illegitimate benefit (e.g., pushing some targeted items into the top-K list of users for market penetration) [**?**]. Later, **?**] first leveraged machine learning methods to propose attacks on factorization-based recommendation systems, which applies the adversarial learning paradigm to generate poisoning input data. Recently, machine-learned adversarial attacks against recommendation systems have received great attention [**? ?**] and numerous works [**? ?**] have reported the failure of machine-learned recommendation models. For example, **?**] showed that the value of nDCG is decreased by -21.2% by exposing the model parameters of BPR [**?**] on adversarial perturbations of the BPR model parameters.

Besides, adversarial attacks have been widely studied in the context of image retrieval. For example, Yang et al. [57] degraded the ranking quality by maximizing the Hamming distance to its own embedding. Chen et al. [7] proposed a query-based black-box attack against image retrieval models to subvert the top-k retrieval results. **?**] proposed to generate retrieval-against universal adversarial perturbations to attack the image retrieval system. The universal adversarial perturbations has been proved could fool deep learning models on most of the data samples. **?**] and **?**] proposed a data-centric proactive privacy-preserving learning algorithm for hashing based retrieval, which utilizes a generator to transfer the original data into the adversarial data with quasi-imperceptible perturbations before releasing them to achieve the data privacy protection. They leverages the adversarial attack to conduct the privacy protection for image retrieval while we propose an adversarial attack for text ranking models. Zhou et al. [60] designed a triplet-like objective function, and combined it with PGD to efficiently obtain the desired adversarial perturbation. In this work, we adopt the PGD to perturb the embedding space according to the expected ranking order.

In text ranking, Raval and Verma [47] explored to lower the rank of a document by token changes. Recently, Goren et al. [16] proposed to promote the rank of a document by replacing a passage in it with some other passages. However, their evaluation for content-quality maintenance highly depends on the human judges, and their study is conducted on feature-based learning to rank models. In this work, we propose automatic evaluation metrics to facilitate the evaluation and study the adversarial attack against prevalent NRMs.

3 PROBLEM STATEMENT

In the Web, there exists an on-going ranking competition: authors of some Web pages may manipulate their documents so as to have them ranked high for many queries [15]. While the traditional retrieval is performed on a relatively static corpus snapshot, nowadays the Web environment becomes competitive [**?**]. Since more and more NRMs are deployed into the real-world applications, the competitive effect needs to be considered for designing NRMs to better fit practical search scenarios. However, there has been little attention paid to consider this effectiveness on NRMs. So we make the first attempt to propose the adversarial attack on NRMs to simulate this real world competitive search [**?**].

In the follows, we will introduce the Word Substitution Ranking Attack (WSRA) task against NRMs, and then describe different adversarial attack settings for the WSRA task.

3.1 Task Description

Typically, given a query q and a set of document candidates $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ selected from a document collection C ($\mathcal{D} \subseteq C$), a ranking model f aims to predict the relevance score $\{f(q, d_n)|n = 1, 2, \ldots, N\}$ between every pair of query and candidate document for ranking the whole candidate set. For example, the ranking model outputs the ranked list $L = [d_N, d_{N-1}, \ldots, d_1]$ if it determines $f(q, d_N) > f(q, d_{N-1}) \cdots > f(q, d_1)$.

Based on these, the WSRA task aims to fool the NRMs to promote a target document in rankings by replacing important words in its text with their synonyms in a semantic-preserving way. In particular, we assume that the attacker is inclined to select $\mathcal{D}$ from the top ranked documents, as the ranked lists returned to the clients are usually "truncated" (i.e., only the partial top-ranked documents will be shown).

In fact, to promote a target document in ranking, there exist multiple ways to design the imperceptible perturbation to the document, e.g., (1) character-level modifications; (2) deleting, adding, or swapping words, and (3) word substitution using semantically similar words. The first two ways are likely to break the grammaticality and naturality of the original input document, and thus can be easily detected by spell or grammar checker [**?**]. In contrast, the third way substitutes

words with semantically similar words, which can preserve semantic consistency and language fluency to the most considerable extent and is often indistinguishable from legitimate ones for human observers [**?**]. Therefore, such word substitutions is a fundamental stepping stone towards identifying the vulnerability of ranking models and helping improve the robustness, which is the focus of this work. That is, the WSRA task aims to promote a target document in rankings by replacing important words in its text with their synonyms.

In this paper, the imperceptibility is reflected in two aspects. Firstly, the adversarial document should be semantic similar to the original document. Secondly, as a feature of adversarial attacks in IR, the adversarial document should easily escape the spam detection. As a verification, we also asked human judges to qualitatively evaluate the imperceptibility.

Formally, given an original target document d, the goal of an attack is to generate a valid adversarial example d^{adv} in the vicinity of d that is ranked higher by NRMs. Specifically, d^{adv} is crafted to conform to the following requirements, i.e.,

$$Rank_L(q, d^{adv}) < Rank_L(q, d) \text{ such that } \mathrm{Sim}(d, d^{adv}) \geq \epsilon, \tag{1}$$

where the adversarial example d^{adv} can be regarded as $d + p$, and p denotes the perturbation to d. $\mathrm{Rank}_L(q, d)$ and $\mathrm{Rank}_L(q, d^{adv})$ denote the position of the original d and its adversarial example d^{adv} in the ranked list L with respect to the query q, respectively. A smaller rank position value represents a higher ranking.

Sim refers to the similarity function between the original d and its adversarial example d^{adv}, and ϵ is the minimum similarity. In the field of natural language, the universal sentence encoder [USE, 5] is often leveraged as the similarity function Sim. USE first maps the two inputs into vector using Transformer encoder, and then computes their cosine similarity as the semantic similarity [23, 27**?**].

Note we can find clear differences between the WSRA task and adversarial attacks in image retrieval and text classification: (1) The WSRA task needs to ensure that the perturbed document is semantically consistent with the original document by imposing a semantic similarity constraint, while the attack against image retrieval makes the pixel-level perturbations bounded in the budget. In essence, continuous image data is tolerant of perturbations to some extent while discrete text data is not [14]; and (2) The WSRA task needs to promote the rank positions in a partial retrieved list, instead of misclassifying the single adversarial sample as in text classification. In this way, existing adversarial attacks against text classifiers for misclassification are incompatible with text ranking models, and we need to thoroughly study the WSRA task to promote the rank positions in a partial retrieved list.

Specifically, in this work, we choose the fine-tuned BERT model on downstream search tasks for adversarial ranking attack, due to the following: (1) the pre-trained language model BERT has shown good superiority on many text ranking problems [17, 24, 29, 32] in both academia and industry in recent years; and (2) previous studies have shown that it is challenging to adversarially attack a fine-tuned BERT on downstream tasks due to its strong performance [23].

3.2 Attack Setting

Attacks that cause the neural ranking model to purposefully promote a target document in the ranking come in two kinds:

White-box: Under the white-box setting, the target model can be fully accessed by attackers. The attackers can directly obtain the real gradient of the loss for the gradient-based attack, which is often conducted by optimizing an attack objective function [**?**].

Black-box: Compared with the white-box attack, the black-box attack is more realistic, since no model information (e.g., parameters and gradients) is available for attackers in reality. The

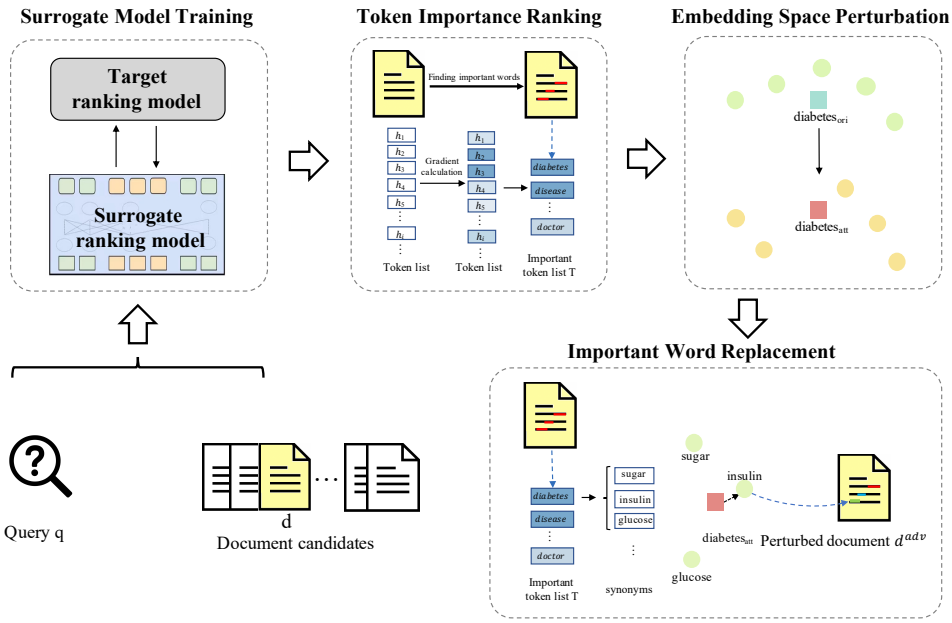

Fig. 2. The overall architecture of the PRADA method. We first query the target NRM to learn a surrogate ranking model based on the PRF idea. Then, we select the important words in a document based on the surrogate model. We apply PGD to generate gradient-based adversarial perturbations to the embedding space towards the expected ranking order. Finally, we iteratively replace the important words with synonyms to find the final adversarial sample.

attackers can only query the target model to achieve the corresponding output. Generally, we can divide black-box attacks into score-based attacks and decision-based attacks.

Score-based: "Score-based" means that the attacker could leverage the relevance score of each candidate document with respect to the query to conduct the attack.

Decision-based: While attackers can still obtain the relevance score under the score-based setting, only the final decision (i.e., rank positions of the partially retrieved list) could be accessed by attackers under the decision-based setting. Therefore, the decision-based setting is more challenging.

In this work, we focus on the *decision-based black-box attack setting* for the WSRA task. Although this setting is significantly more challenging than white-box and score-based black-box attacks to NRMs, it is more practical and enables to apply our methods to attack a real-world search engine. Since more and more NRMs are deployed into the real-world applications, this adversarial attack will reveal the vulnerabilities of real-world search engines and enlighten the design of more robust NRMs in the future.

4 OUR ATTACK METHOD

In this section, we introduce our proposed attack method for the WSRA task under the decision-based black-box attack setting. We first give an overview of the model architecture and then describe each component of the model in detail.

4.1 Model Overview

In this work, we formulate the attack goal of the WSRA task that promotes the target document d in rankings with respect to a query $q \in Q = \{q_1, \ldots, q_{|Q|}\}$ by perturbation p as the following

Algorithm 1 PRADA

Inputs: a query q, a pre-collected query collection Q_C, a set of document candidates $\mathcal{D}$, a target ranking model f, a target document d
Output: an adversarial document d^{adv}

Procedure Surrogate Model Training
for $q_c \in Q_C$ **do**
  Get the ranked list $\boldsymbol{L}_c$ by querying the target model with q_c.
end for
Train the surrogate model f_s in terms of Eq.(6).
Procedure Token Importance Ranking
$H = \{h_1, h_2, ..., h_i, ...\}$ // sub-word token list of d
Compute the importance score I_{h_i} for each h_i in terms of Eq.(7).
Rank H in descending order to create $T[:m]$
Procedure Embedding Space Perturbation
for $t \leftarrow 1$ to η **do**
  Compute the gradient $\boldsymbol{g}_{d_t^{adv}}$ of Eq. (4) using Eq.(8)
  Update the adversarial candidate d_{t+1}^{adv} in terms of Eq.(9)
end for
Obtain the perturbed vectors $\mathbf{o}^p$ of the m important tokens
procedure Important Word Replacement
Initialization: $d^{opt} \leftarrow d$
for $o_i \in T[:m]$ **do**
  Find the corresponding whole word w_{o_i}, $\boldsymbol{e}_{cf}(w_{o_i}) \leftarrow \text{map}(w_{o_i})$
  Obtain the S synonyms $\{w_s\}_{s=1}^{S}$ in terms of Eq.(10)
  $\mathbf{e}_{w_s} \leftarrow \text{encode}(w_s)$
  $w_s* = \text{argmax}_{w_s \in \{w_s\}_{s=1}^{S}} \text{CosSim}(\mathbf{e}_{w_s}, \boldsymbol{e}_{cf}(w_{o_i}))$
  if $Rank_L(q, d^{temp}) < Rank_L(q, d^{opt})$ **then**
    $d^{opt} \leftarrow d^{temp}$
  end if
end for
return $d^{adv} = d^{opt}$

problem:

$$p = \arg\min Rank_L(q, d + p). \tag{2}$$

The optimization problem cannot be directly solved due to the discrete nature of the rank position $\text{Rank}_L(q, d)$. To solve the problem, we design a surrogate objective function following [60]. The attacking goal in Eq. (2) can be converted into a series of inequalities, i.e.,

$$Rank_L(q, d + p) < Rank_L(q, L_{\setminus d}), \tag{3}$$

where d and $L_{\setminus d}$ denote the target document and the remaining documents from the ranked list L, respectively. Each inequality represents a pairwise ranking sub-problem between d and other documents $L_{\setminus d}$. The adversarial candidate $d^{adv} = d + p$ should be ranked ahead of other documents with respect to q.

Here, we leverage the pairwise hinge loss to model the expected ranking order, i.e.,

$$L_{RA}(q, d + p; L) = \sum_{d' \in L_{\setminus d}} \max(0, \beta - f_s(q, d + p) + f_s(q, d')), \tag{4}$$

where β is the margin for the hinge loss function, which is often set to 1, and f_s denotes the relevance score given by the surrogate ranking model, which will be described next; d' denotes the remaining documents in the ranked list L without the target document d.

In this way, the original problem in Eq. (2) can be reformulated into the following optimization problem:

$$p = \arg\min L_{RA}(q, d + p; L). \tag{5}$$

To ensure the quality of the adversarial examples that are being generated, we further impose constraints on the ranking attack on the following three aspects: (1) the maximum number of modified tokens in a document, m, (2) the maximum number of one word's synonyms, S, and (3) the minimum semantic similarity between the original target document and the adversarial example, ϵ.

To solve the optimization problem in Eq. (5) and satisfy the required constraints, we introduce a novel Pseudo-Relevance based ADversarial ranking Attack method, or PRADA for short. The overall architecture of PRADA is depicted in Figure 2. A pseudo algorithm for PRADA is provided in Algorithm 1.

Briefly, PRADA can be decomposed into four dependent components: (1) Surrogate Model Training, to learn a surrogate model that can imitates the behaviors of the target NRM based on the PRF idea; (2) Token Importance Ranking, to find the important words in the document that have a strong influence on the rankings; (3) Embedding Space Perturbation, to generate the desired adversarial perturbation in the embedding space for the important words; and (4) Important Word Replacement, to iteratively replace the important words one by one based on their perturbed vectors and synonyms to find adversarial samples that can mislead the target model. Below, we discuss each of the components.

4.2 Surrogate Model Training

In adversarial attacks, the gradients for guiding the attack process are usually calculated based on knowledge of the target model, which is unavailable under the black-box setting. Hence, based on the PRF idea in IR, we propose to train a surrogate ranking model [41, 42] with similar behaviors of the target model. Then, we can obtain prior-guided gradients, and attack the target ranking model based on the surrogate model due to the transferability [41].

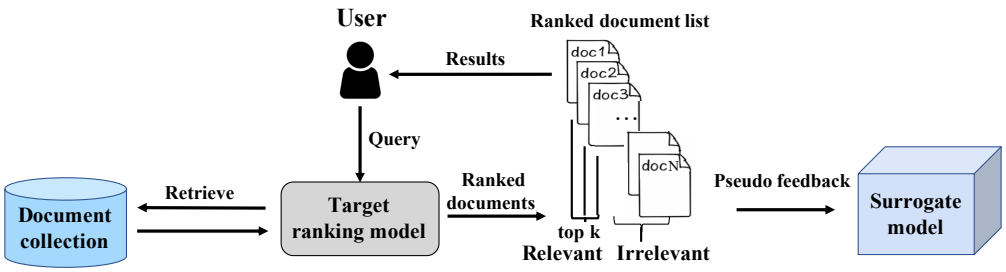

Fig. 3. The training process for the surrogate model.

As shown in Figure 3, given a random query q_c from a pre-collected query collection Q_C, the target model returns a ranked list L_c with N documents. To obtain the priors for attacks, we query the target model with all $|Q_C|$ queries collected from the downstream search tasks. We generate pseudo-labels as the ground-truth by treating the top-k ranked documents as relevant while treating the other documents as irrelevant, for training the surrogate ranking model f_s. The objective function is defined as

$$L_s = \frac{1}{|Q_C|} \sum_{c=1}^{|Q_C|} \max(0, \beta - f_s(q_c, L_c[:k]) + f_s(q_c, L_c[k+1:N])), \tag{6}$$

where $L_c[:k]$ denotes the top k ranked documents, and $L_c[k+1:N]$ denotes the remaining documents in the list; β is the margin for the hinge loss function, which is often set to 1.

Specifically, for the surrogate model, we initialize it using the original BERT since it can achieve substantial performance improvements in recent IR studies [17, 32**?**]. For the target attack model, we choose the fine-tuned BERT as discussed in Section 3.1. Meanwhile, to further verify the effectiveness of our method when the surrogate model is dissimilar to the attack model, we also leverage our method with the surrogate model as BERT, to attack other target NRMs, e.g., Conv-KNRM [**?**] and Duet [36]. The experimental results are in Section 6.5, which show that our method can still work well under this setting.

4.3 Token Importance Ranking

Given a target document d, which is tokenized into sub-word token list $H = [h_1, h_2, \ldots, h_i, \ldots]$ by BERT, we observe that only some important tokens act as influential signals for the surrogate ranking model f_s, echoing the observation in [23] that BERT attends to the statistical cues of some words. That is, perturbations over these important tokens can be most beneficial in crafting adversarial samples. Therefore, we propose a scoring mechanism to identify the important tokens in a document which have a high impact on the final ranking result.

Following [26, 55], we first calculate the gradient magnitude with respect to each input unit. Then, we sum up the score for each dimension in the embedding space as the token-level importance score. Specifically, we introduce a scoring function that determines the importance I_{h_i} of the i-th token h_i in d as

$$I_{h_i} = \left\| \frac{\partial L_{RA}}{\partial \boldsymbol{e}^o_{h_i}} \right\|_2^2, \tag{7}$$

where $\boldsymbol{e}^o_{h_i}$ is the original embedding vector of h_i in the surrogate model; L_{RA} denotes the adversarial ranking objective function, which is defined in Eq. (4).

We rank all the tokens according to the importance score I_{h_i} in descending order to create the candidate token list T. We only attack the top m important tokens for each d, i.e., $T[:m] = [o_1, o_2, \ldots, o_m]$, since we intend to keep the perturbation to a minimum.

4.4 Embedding Space Perturbation

NRMs usually map samples (i.e., queries and documents) to an embedding space, where the distances among them determine the final ranking order [60]. A document's position in the embedding space may be changed by adding a perturbation to its important tokens. Therefore, we generate gradients based on the surrogate model for finding a proper perturbation to the important tokens, which could push the document to a desired position.

Specifically, we adopt the Projected Gradient Descent [PGD, 33] method, which is one of the most effective first-order gradient-based algorithms. Note that in this work, the perturbation p is achieved at the token-level instead of at the document-level.

For all m important tokens $o_i \in T[:m]$, the PGD algorithm alternates two steps at every iteration $t = 1, 2, \ldots, \eta$:

- Calculating the gradient $\boldsymbol{g}_{d_t^{adv}}$ of Eq. (4), i.e.,

$$\boldsymbol{g}_{d_t^{adv}} = \frac{\partial L_{RA}(q, d_t^{adv}; L)}{\partial d_t^{adv}}, \tag{8}$$

  where d_t^{adv} denotes the adversarial document with the embedding of all the important tokens perturbed at the t-th step.

- Leveraging the gradient $\boldsymbol{g}_{d_t^{adv}}$ to update the adversarial candidate, i.e.,

$$d_{t+1}^{adv} = d_t^{adv} - \alpha \frac{\boldsymbol{g}_{d_t^{adv}}}{\left\| \boldsymbol{g}_{d_t^{adv}} \right\|_2^2}, \tag{9}$$

where α denotes a constant hyper-parameter indicating the PGD step size and d_1^{adv} is initialized as the original d. Note that we removed the clip operation in the original PGD algorithm since we have found that it limits the perturbation in the embedding space, which leads to poor experimental results.

After η iterations for all the important token o_i, we obtain the final perturbed vectors of the m important tokens $T[:m]$, i.e., $\mathbf{o}^p = \{\boldsymbol{e}_{o_1}^p, \boldsymbol{e}_{o_2}^p, \ldots, \boldsymbol{e}_{o_m}^p\}$.

4.5 Important Word Replacement

Based on the perturbed vectors of m important tokens $T[:m]$, we replace the important token with semantically similar and grammatically correct words and repeat this process by iterating the list $T[:m]$ to find the final adversarial sample. Specifically, we generate a set of synonyms for each important token for replacement, to satisfy the requirement of semantic similarity in Eq. (1).

For a target document d, the word replacement phase includes the following steps:

Extracting synonyms for each important token. For each important token $o_i \in T[:m]$, we first find its corresponding whole word w_{o_i}. If o_i is a single word, the corresponding whole word is itself. Otherwise, we search back and forth to recover the corresponding whole word. Then, w_{o_i} is mapped into the counter-fitted word embedding space [37] where only synonyms are close to each other, to obtain the word vector $\boldsymbol{e}_{cf}(w_{o_i})$. For each $\boldsymbol{e}_{cf}(w_{o_i})$, we obtain the top S synonyms $\{w_s\}_{s=1}^{S}$ via

$$\text{Sim}(\boldsymbol{e}_{cf}(w_{o_i}), \boldsymbol{e}_{cf}(w_s)) \geq \lambda, \tag{10}$$

where Sim denotes the cosine similarity between two counter-fitted embeddings, w_s denotes the synonym of w_{o_i}, and λ denotes the minimum similarity between w_{o_i} and w_s. Furthermore, for each synonym w_s with respect to w_{o_i}, we encode it back to the embedding space of the surrogate model to obtain the embedding $\mathbf{e}_{w_s}$. Note that if the synonym is tokenized by BERT, $\mathbf{e}_{w_s}$ is obtained by the average of sub-word token embeddings.

Greedy word replacement strategy. We calculate the cosine similarity between the candidate synonym vector $\mathbf{e}_{w_s}$ and the corresponding perturbed word vector $\boldsymbol{e}_{o_i}^p \in \mathbf{o}^p$. The synonym w_s^* which has the highest cosine similarity with w_{o_i} is chosen to replace w_{o_i}. Suppose the document before this word replacement process is $d^{opt} = \{w_1, w_2, \ldots, w_{o_i}, \ldots\}$, the document after the word replacement is $d^{temp} = \{w_1, w_2, \ldots, w_{o_{i-1}}, w_s^*, w_{o_{i+1}} \ldots\}$. Simply replacing a token by its synonym cannot guarantee a successful attack. Therefore, we adopt a greedy word replacement strategy. Specifically, we obtain the rank of d^{temp} by querying the target model. If the rank of d^{temp} has improved, i.e., $Rank_L(q, d^{temp}) < Rank_L(q, d^{opt})$, we accept the replacement and denote d^{opt} as the d^{temp}, i.e., $d^{opt} \leftarrow d^{temp}$. Otherwise, we will discard this word replacement and turn to the next round.

The process described above is repeated by iterating over the importance word list $T[:m]$ to find the final adversarial sample d^{adv}.

5 EXPERIMENTAL SETUP

In this section, we introduce our experimental settings.

5.1 Datasets

To evaluate the effectiveness of our proposed methods, we conducted experiments on two web search benchmark datasets.

- **MS MARCO Document Ranking dataset** [38] (MS-MARCO-Doc) is a large-scale benchmark dataset for web document retrieval, with about 3.21 million web documents.
- **MS MARCO Passage Ranking dataset** [38] (MS-MARCO-Pas) is a large-scale benchmark dataset for passage retrieval, with about 8.84 million passages from web pages.

Table 1. Data statistics. #w denotes the number of words.

	MS-MARCO-Doc	MS-MARCO-Pas
Training queries	0.37M	0.5M
Dev queries	5,193	6,980
Documents/passages	3.21M	8.84M
Documents/Passages: avg #w	1,129	58

Detailed dataset statistics are shown in Table 1. We take these datasets for experiments since (1) Relevant documents for each user's query are retrieved using Bing from its large-scale web index, which is representative of real web search scenario. (2) It is practical to promote irrelevant documents instead of relevant documents in rankings. The probability of selecting relevant document for attack is low since each query has only one relevant document.

5.2 Baselines

We adopt two types of baselines for comparison, including step-wise methods and traditional term spamming methods.

5.2.1 Step-wise Methods. For step-wise methods, we apply two steps to attack the target document, where the first step is to select n words in the document, and the second step is to substitute these words. For the word selection step, we employ four methods:

- **First** selects the first n words in the document to attack.
- **Last** selects the last n words in the document to attack.
- **Tf-idf** selects the top n words with the highest tf-idf scores in the document to attack.
- **TextRank** selects n words by TextRank [34], a graph-based method inspired by the PageRank algorithm.

For the word replacement step, we employ two methods:

- **Random Replacement (RR)** replaces the selected word with a random word.
- **Nearest Replacement (NR)** replaces the selected word with the nearest word in the Glove [43] using cosine similarity.

By combining these two-step methods, we obtain eight types of attack methods denoted as **First+RR**, **First+NR**, **Last+RR**, **Last+NR**, **Tf-idf+RR**, **Tf-idf+NR**, **TextRank+RR**, and **TextRank+NR**.

5.2.2 Traditional Term Spamming Methods. Term spamming [19] refers to techniques that tailor the contents of a web page's text fields to rank it higher than they deserve. Here, we apply two traditional term spamming methods:

- **Repetition ($\mathbf{TS}_{Rep}$)** promotes the rank of d by adding a small number of query terms [19]. We randomly choose a starting position in d and replace the following successive n words with n query terms.
- **Stitching ($\mathbf{TS}_{Sti}$)** is to manually glue together sentences from other documents [19]. We randomly choose a starting position in d and replace the following successive n words with n words extracted from a sentence pool S_{pool}. Following [16] where authors tend to mimic content in documents that were highly ranked in the past for a query of interest, we construct S_{pool} by collecting sentences in documents that are ranked higher than d.

5.3 Evaluation Metrics

For the evaluation of the WSRA task, we set up various **automatic evaluation metrics** and **human evaluation metrics**, respectively.

5.3.1 Automatic Evaluation Metrics. For automatic evaluation metrics, we first set up the Success Rate (SR) metric to measure the document ranking attack effect, i.e.,

Success Rate (SR) evaluates the percentage of the after-attack documents that are ranked higher than original documents. We define SR as

$$\text{SR} = \frac{1}{|Q|}\sum_{t=1}^{|Q|}\frac{1}{N_q}\sum_{i=1}^{N_q}\mathbb{I}\{Rank_L(d_i+p) < Rank_L(d_i)\},$$

where $|Q|$ denotes the number of evaluated queries, N_q the number of attacked documents with respect to each query, and d_i the attacked document with respect to the query $q \in Q$. $\mathbb{I}\{\cdot\}$ is the indicator function. The effectiveness of an adversarial attack is better with a higher SR (%).

Furthermore, we adapt the Perturbation Percentage (PP) and Semantic Similarity (SS) from NLP to measure the quality of the generated samples:

Perturbation Percentage (PP) evaluates the word-level perturbation percentage of candidate documents following [27]. Specifically, it will compare each word in the adversarial document with the original document to see how many words have been changed. The PP is the number of changed words divided by the total number of words in the document. A lower PP (%) results in higher semantic consistency.

Semantic Similarity (SS) evaluates the semantic similarity between the original document and the adversarial example. Following [23, 26], we use the USE to measure the semantic similarity. We set the encoding model to the released deep averaging network[1] since it can encode long documents quickly. In this work, we evaluate SS at both the document-level ($\mathbf{SS}_{doc}$) and sentence-level ($\mathbf{SS}_{sen}$). For SS_{doc}, we directly input two documents and evaluate the semantic similarity between them. For SS_{sen}, we first split two documents into sentence pairs and then evaluate the average sentence semantic similarity between these sentence pairs. A higher SS (%) results in higher semantic consistency.

5.3.2 Human Evaluation Metrics. Besides the automatic evaluation metrics, we further conduct human evaluations to measure the quality of the attacked documents from three aspects: (1) fluency in grammar; (2) imperceptibility to human judges; and (3) semantically consistency with original documents.

We first randomly sample 40 test queries from MS-MARCO-Doc and take the corresponding 9 original documents for each query. Then, we find the 360 adversarial samples generated by PRADA and TR_{rep}, respectively. We shuffle a mix of original and adversarial documents (i.e., 1,080 in total) and asked three labelers to evaluate them. For (1), annotators score the quality of the mixed examples from 1–5 following [27]. For (2), annotators judge each example whether it is attacked (i.e., labeled as 0) or not (i.e., labeled as 1). For (3), we compare adversarial samples generated by PRADA and TR_{rep} with the original documents, using the following criteria: i) 2: the adversarial sample is completely semantically consistent with the original document; ii) 1: the adversarial sample is partially relevant with the original document and human can still understand the original information; and iii) 0: the adversarial sample is not relevant with the original document. Note that for (3), we conduct a separate evaluation after (1) and (2). We first mix adversarial documents generated by PRADA and TR_{rep}. For each adversarial document, we give the human judge its original document as a reference to obtain the semantic consistency between these two documents.

[1] https://tfhub.dev/google/universal-sentence-encoder/2

Agreements to measure inter-rater consistency among three labelers are calculated with the Fleiss' kappa [12].

5.4 Implementation Details

In the surrogate model training process: (1) For the target attack model, we obtain it by fine-tuning BERT on the training queries of the MS-MARCO-Doc and MS-MARCO-Pas, respectively. Following [9], we apply BERT_{base} released by Google. Besides, to verify the effectiveness of our method to attack other NRMs (i.e., Section 6.5), we choose the Conv-KNRM [**?**] and Duet [36] as the target model which are implemented and trained following the previous work [**?**]. (2) For the surrogate ranking model, we initialize it using the original BERT. To train it, we leverage the test queries of the MS-MARCO-Doc and MS-MARCO-Pas as Q_c, respectively. Following [9], we apply BERT_{base} released by Google. For the MS-MARCO-Doc, we use the official top 100 ranked documents retrieved by the QL model following [8]. For the MS-MARCO-Pas, initial retrieval is performed using the Anserini toolkit [58] with the BM25 model to obtain the top 100 ranked passages following [31]. The ranked list L_c is obtained by utilizing the target ranking model to re-rank the above initial ranked list and the length N is set to 100. We set $k = 1$ in Eq. (6) since every query in the MS-MARCO-Doc and most queries in the MS-MARCO-Pas have only one relevant document.

In the token importance ranking process, the number of top important tokens m in PRADA is set to 50 and 20 for the MS-MARCO-Doc and MS-MARCO-pas, respectively. For fair comparison with the baselines, we also set n to 50 and 20 for the MS-MARCO-Doc and MS-MARCO-pas, respectively. Besides, we will analyze the effect of m in PRADA on the attack performance.

In the embedding space perturbation process, the PGD step size α is set to 45 and the number of iteration η is set to 3. In the important word replacement process, we set the minimum similarity λ to 0.5.

We evaluate PRADA on 200 queries (i.e., $|Q| = 200$) randomly sampled from the dev set in the MS-MARCO-Doc and MS-MARCO-Pas datasets, respectively. For each query, we attack 9 target documents in the top 100 documents, which are obtained by picking 1 out of every 10 documents. Specifically, we randomly choose 1 document from 9 ranges in the document list, i.e., [11, 20], [21, 30], …, [91, 100], respectively. Note that we do not choose documents from the range of [1,10] since it is not necessary to attack the top-10 documents for ranking promotion.

6 EXPERIMENTAL RESULTS

In this section, we report and analyze the experimental results to demonstrate the effectiveness of the proposed PRADA method. Specifically, we target to answer the following research questions:

- **RQ1**: How does PRADA perform compared with baselines under the automatic and human evaluations?
- **RQ2**: Can PRADA evade detection by an anti-spamming method?
- **RQ3**: How do different components of the PRADA affect the performance?
- **RQ4**: How does PRADA perform under the white-box attack setting?
- **RQ5**: How does PRADA perform when attacking different NRMs?
- **RQ6**: How does PRADA perform for different rank positions in the document list?
- **RQ7**: How does the number of important tokens m affect the PRADA performance?

6.1 Baseline Comparison

To answer **RQ1**, we compare PRADA with different baselines under both the automatic evaluations and human evaluations.

Table 2. Comparisons between PRADA and the baselines under the automatic evaluation; * indicates significant improvements over the next-best approach (two-tailed t-tests, p-value < 0.05).

Method	MS-MARCO-Doc				MS-MARCO-Pas			
	SR	PP	SS_{doc}	SS_{sen}	SR	PP	SS_{doc}	SS_{sen}
First+RR	65.9	13.0	90.9	92.0	9.3	24.5	78.1	79.7
First+NR	41.9	13.0	94.3	94.9	14.8	24.5	85.9	86.3
Last+RR	10.7	13.0	91.1	91.9	20.7	24.5	78.7	81.5
Last+NR	8.2	13.0	94.7	95.1	22.7	24.5	86.2	87.1
Tf-idf+RR	48.1	13.0	90.5	90.5	8.8	24.5	80.5	80.5
Tf-idf+NR	43.8	13.0	93.1	93.0	10.1	24.5	81.5	81.1
TextRank+RR	55.4	13.0	87.4	88.8	8.7	24.5	74.2	73.7
TextRank+NR	37.5	13.0	90.8	92.7	13.9	24.5	84.2	83.6
TS_{rep}	93.1	12.8	87.9	89.1	**99.5**	24.0	85.6	87.5
TS_{sti}	70.9	12.9	91.2	91.7	59.9	24.3	86.8	87.0
PRADA	**96.7***	**4.0***	**95.2***	**96.2***	91.4	**7.8***	**93.2***	**93.1***

Automatic evaluation. The performance comparisons between our model and the baselines are shown in Table 2. For the MS-MARCO-Doc, we have the following observations: (1) Step-wise methods generally perform worse than term spamming methods and PRADA in terms of SR, indicating that promoting the document in rankings is a non-trivial problem. (2) For step-wise methods, the methods based on NR perform better than that based on RR in terms of SS_{doc} and SS_{sen}. (3) Term spamming methods perform the best in terms of SR among the baselines. TS_{rep} performs better than TS_{sti}, indicating that replacing words in a document with the query terms is better than words from other documents. (4) PRADA performs best in terms of all the automatic evaluation metrics. That is, PRADA achieves a high success rate while maintaining a minimum perturbation, indicating the perturbation of the important words would be easier to result in ranking promotion from the target model, which is consistent with previous observations [27].

When we look at the performance of different models on the MS-MARCO-Pas, we find that PRADA performs worse than TS_{rep} in terms of SR when dealing with short texts. A potential reason is that there are few options in the passage for determining important tokens. This result is consistent with the previous observation [27], where the sequence length plays an important role in the high-quality perturbation process and the word replacement would be less reasonable when dealing with extremely short sequences. For TS_{rep}, it directly adds query words into the target document to promote its ranking. This way, it will contribute to rank promotion significantly for short texts. By analyzing the MS-MARCO-Pas dataset, we found that in average, there are about 24% words in a perturbed document generated by term spamming are query words. When applying this method to neural ranking models, each added query word contributes significantly to the rank promotion since the exact matching signal is of great importance By conducting further analysis, we find that PRADA prefers to keep the original words due to the unsuccessful attack under the greedy word replacement strategy, which contributes to the semantic consistency. In future work, we aim to consider a more advanced objective towards both long text and short text for developing robust NRMs.

Human evaluation. Table 3 shows the human evaluation results. Note that the Semantic consistency is evaluated between the adversarial documents and the original documents. So there is

Table 3. Comparisons between PRADA and TS_{rep} under the human evaluation.

	Grammar	kappa	Imperceptibility	kappa	Semantic	kappa
Original	3.50	0.373	0.88	0.475	-	-
TS_{rep}	1.69	0.177	0.06	0.647	0.43	0.298
PRADA	3.23	0.478	0.85	0.486	1.37	0.412

no "Semantic" and "kappa" for "Original". We can observe that (1) The semantic consistency and language fluency of the adversarial examples generated by PRADA are better than that generated by TS_{rep}. The adversarial examples generated by PRADA are more imperceptible to human judges than TS_{rep}. Intuitively, humans can easily identify an attacked document with multiple successive repetitive words. All the human judgement results again demonstrate the effectiveness of our PRADA method. (2) The kappa values of PRADA for all three aspects are larger than 0.4, considered as "moderate agreement" regarding quality of adversarial examples. The largest kappa value (i.e., 0.647) is achieved by TS_{rep} for imperceptibility, which seems reasonable since it is easy to reach an agreement on the attacked documents with successive repetitive words.

6.2 Spam Detection

To answer **RQ2**, we adopt the representative utility-based term spamicity method [59], which can online detect whether target pages are spam or not, to detect the adversarial examples generated by PRADA and the best baseline model TS_{rep} on the MS-MARCO-DOC. Specifically, if the spamicity score is higher than a utility threshold β, such example is detected as a spam. We vary the threshold β by setting it to seven different values (i.e., 0.080, 0.075, 0.070, 0.065, 0.060, 0.055, 0.050). The results of detection rates are shown in Table 4. We have three main observations: (1) The detection rate increases with the decrease of the threshold β. (2) TS_{rep} can be very easily detected under the spam detection algorithm since it puts many query terms into documents. (3) PRADA outperforms TS_{rep} significantly (p-value < 0.05). It is much easier for PRADA to evade the spam detection (e.g., for $\beta = 0.050$, the detection rate of PRADA and TS_{rep} is less than 20% and over 99%, respectively).

Table 4. The detection rate (%) of PRADA and TS_{rep} via a representative anti-spamming method; * indicates statistically significant improvements over TS_{rep} (two-tailed t-tests, p-value < 0.05).

β	0.080	0.075	0.070	0.065	0.060	0.055	0.050
TS_{rep}	81.0	85.8	90.2	93.4	96.2	98.2	99.4
PRADA	7.1*	8.2*	9.3*	11.4*	13.9*	15.6*	19.2*

6.3 Model Ablation

To answer **RQ3**, we conduct an ablation analysis to investigate the effect of proposed different components in our PRADA method. We implement several variants of PRADA by removing major components, and adopting different strategies:

- **PRADA**$_{-TIR}$ removes the step of finding important words described in Section 4.3, and randomly selects words to attack.
- **PRADA**$_{-ESP}$ removes the embedding space perturbation described in Section 4.4, and applies random perturbations on the embedding space.
- **PRADA**$_{-IWR}$ removes the important word replacement described in Section 4.5, and replaces all important words in the document with words that are nearest to the perturbed word vectors.
- **PRADA**$_{-ESP-IWR}$ removes both the embedding space perturbation and word replacement. It applies random perturbations on the embedding space and directly selects the nearest word to replace the important word.

Based on Table 5, we observe that: (1) By removing important word replacement, the performance of PRADA$_{-IWR}$ in terms of SR has a significant drop as compared with PRADA. The results indicate that the greedy synonym replacement strategy does help the rank promotion. PRADA$_{-ESP}$ has a similar performance with PARDA, which again demonstrates the effectiveness of the word replacement with synonyms. (2) PRADA$_{-ESP-IWR}$ performs much worse than the PRADA$_{-IWR}$. Without the limitation given by the word replacement, the embedding space perturbation has an obvious influence on the results. (3) By including all the components, PRADA achieves the best performance among the variants in terms of all evaluation metrics.

Table 5. Model analysis of PRADA under automatic evaluations; * denotes significant degradation w.r.t. PRADA (two-tailed t-tests, p-value<0.05).

Method	MS-MARCO-Doc				MS-MARCO-Pas			
	SR	PP	SS_{doc}	SS_{sen}	SR	PP	SS_{doc}	SS_{sen}
PRADA$_{-TIR}$	86.1*	4.0	94.8	95.7	87.1*	7.8	92.7	89.9*
PRADA$_{-ESP}$	94.7	4.0	94.6	95.3	90.2	7.8	92.6	89.8*
PRADA$_{-IWR}$	39.6*	4.5	92.5	94.8	47.8*	8.5	82.6*	86.4*
PRADA$_{-ESP-IWR}$	5.8*	4.5	92.3	94.7	10.6*	8.5	82.4*	86.1*
PRADA	**96.7**	**4.0**	**95.2**	**96.2**	**91.4**	**7.8**	**93.2**	**93.1**

6.4 Analysis between Black-box Setting vs. White-box Setting

As mentioned in Section 3.2, there are different adversarial settings for the WSRA task in terms of the information that attackers rely on. In this work, we focus on the decision-based black-box attack setting because it is close to real-world search engine scenario. This setting has been extensively studied in the image domain, but has yet to be explored in the context of IR. While such setting is more challenging than the white-box setting, it is also meaningful to explore the white-box setting to further understand the ranking model's robustness against the WSRA. To this end, we conduct a white-box WSRA on the MS-MARCO-Doc and MS-MARCO-Pas to answer **RQ4**.

Specifically, we firstly show the performance of the surrogate ranking model compared with the target ranking model on two datasets. Then, we compare the attack performance between the black-box setting and the white-box setting under the automatic evaluations.

Table 6. The ranking performance of the surrogate ranking model vs. the target ranking model under the MRR@10 and MRR@100.

Model	MS-MARCO-Doc		MS-MARCO-Pas	
	MRR@10	MRR@100	MRR@10	MRR@100
Surrogate ranking model	0.3471	0.3547	0.3313	0.3376
Target ranking model	0.3813	0.3868	0.3437	0.3490

Surrogate ranking model vs. target ranking model. We evaluate the ranking performance of the surrogate model and target model over all the queries on the dev sets of the MS-MARCO-Doc and MS-MARCO-Pas, respectively. For both datasets, we report the Mean Reciprocal Rank at 10 (**MRR@10** [**?**]) and the Mean Reciprocal Rank at 100 (**MRR@100** [**?**]), as suggested in the official instructions. The results are shown in Table 6. From the results, we can find that: (1) The target ranking model performs well on the MS-MARCO-Doc and MS-MARCO-Pas datasets. (2) The surrogate ranking model performs slightly worse than the target ranking model, because

they are trained on the weakly annotated training set instead of the ground-truth label. (3) The performance gap between the surrogate ranking model and the target ranking model is small. For example, the MRR@10 of the surrogate model is only about 0.01 worse than the target model on the MS-MARCO-Pas. The results indicate that our proposed surrogate model training method is sufficient to mimic the behaviors of the target model, which provides high-quality foundation for the subsequent attack steps in our method.

Table 7. Attack performance comparisons of PRADA between the black-box setting and the white-box setting under the automatic evaluation.

Method	MS-MARCO-Doc				MS-MARCO-Pas			
	SR	PP	SS_{doc}	SS_{sen}	SR	PP	SS_{doc}	SS_{sen}
PRADA	96.7	**4.0**	**95.2**	**96.2**	91.4	**7.8**	**93.2**	**93.1**
White-PRADA	**96.8**	4.4	94.6	95.8	**94.4**	10.2	92.0	91.8

Attack performance comparison. We then compare the attack performance of our PRADA between the black-box setting and the white-box setting under the automatic evaluations. To conduct the white-box WSRA, we directly set the surrogate model as the target model (i.e., both of them are the fine-tuned BERT) and keep other components the same in our method, denoted as White-PRADA. The results are shown in Table 7. From the results, we can observe that: (1) White-PRADA performs better than PRADA in terms of the SR. The major reason is that White-PRADA can have full access to the target ranking model. In this way, the gradient White-PRADA produced for token importance ranking and embedding space perturbation is more precise than PRADA based a surrogate model. The result also shows an upper bound on the SR score that PRADA can achieve. (2) PRADA performs better than White-PRADA in terms of PP, SS_{doc} and SS_{sen}. The reason might be that the important word replacement in PRADA is more difficult to provide rank promotion than White-PRADA, since the gradient information in PRADA is less precise than White-PRADA. Due to the greedy word replacement strategy, PRADA prefers to keep the original words in the target document, which contributes to the semantic consistency.

Table 8. Attack performance comparisons among different neural ranking models under the automatic evaluation on the MS-MARCO-Doc dataset.

Target Model	SR	PP	SS_{doc}	SS_{sen}
BERT	**96.7**	4.0	95.2	96.2
Conv-KNRM	96.1	3.0	96.1	97.0
Duet	95.7	**2.9**	**96.3**	**97.2**

6.5 Analysis of Attacking against Different NRMs

In real-world practice, it is often difficult for the attacker to know the target model. To simulate the practical IR setting, we test the attack performance of our PRADA against different target ranking models. Specifically, we leverage PRADA to attack different NRMs on the MS-MARCO-Doc and MS-MARCO-Pas dataset to answer **RQ5**. Specifically, we take BERT, Conv-KNRM [**?**] and Duet [36] as the target model and keep BERT as the surrogate model. The results are shown in Table 8. We have three main observations: (1) BERT achieves the best SR among different target models, indicating that it is easier to attack when the target model and the surrogate model have the same structure. (2) Using PRADA to attack other NRMs, the success rate is still high (e.g., 96.1 and 95.7 on Conv-KNRM and Duet, respectively). The reason might be that adversarial documents in IR

have similar transferability as that in the computer vision field [41**?**]. That is, adversarial examples generated by a model (e.g., surrogate model BERT) could successfully attack an unrelated model (e.g., target model Conv-KNRM/Duet). Besides, querying the target model in the important word replacement step also plays an important role in making the attack successful.

6.6 Analysis at Different Rank Positions

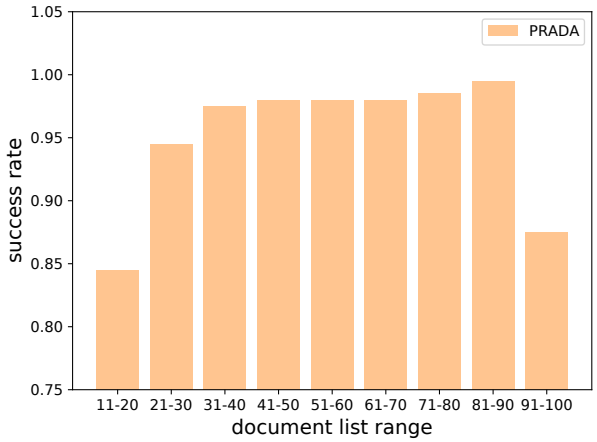

Fig. 4. Success rate at different ranges of the ranked list from PRADA on MS-MARCO-Doc.

To answer **RQ6**, we analyze the success rate for documents with different rank positions on the MS-MARCO-DOC by PRADA. Specifically, we visualize the distribution of the success rate in different ranges of the document list (i.e., [11, 20], [21, 30], ..., [91, 100]) in Figure 4. As we can see: (1) In general, it is harder to promote high-ranked documents in rankings than low-ranked documents. Documents in the range of [11,20] are the most difficult to be attacked, with a SR value as only 0.845. (2) It is surprising to find that documents in the last range (i.e., [91,100]) achieve a low success rate (i.e., 0.875). One possible explanation is that these documents are too irrelevant to be promoted in rankings. It is necessary to focus on the attack against low-ranked documents in the future.

6.7 Analysis of the Number of Important Tokens

To answer **RQ7**, we analyze the effect of different numbers of important tokens m for PRADA on the attack performance. Specifically, we compare PRADA with the best performing baseline TS_{rep} on the MS-MARCO-Doc and set m to six different values (i.e., 10, 20, 30, 40, 50, 60). Note that the selected number n of TS_{rep} is equal to m. As shown in Figure 5, we find that: (1) Overall, the SS and PP increases with the increase of m for both PRADA and TS_{rep}. This result indicates that attacking more words is more likely to promote the rank. (2) Intuitively, a larger m would result in less semantic similarity. The SS_{doc} of TS_{rep} has a larger drop than PRADA in the range of [30,60], and the performance of PRADA in terms of SR and PP is always better than TS_{rep} with different m. These results again illustrate the effectiveness of PRADA.

6.8 Case Study

To obtain a better qualitative understanding of how different models perform, we show the adversarial examples from PRADA as well as that from TS_{rep}, with the number of important tokens m set to 50. We take one query "government does do" from the dev set of the MS-MARCO-Doc as an example. Due to space limitations, we only show some key sentences in the document. As shown in Table 9, we can observe that (1) Compared with TS_{rep}, the adversarial document generated by

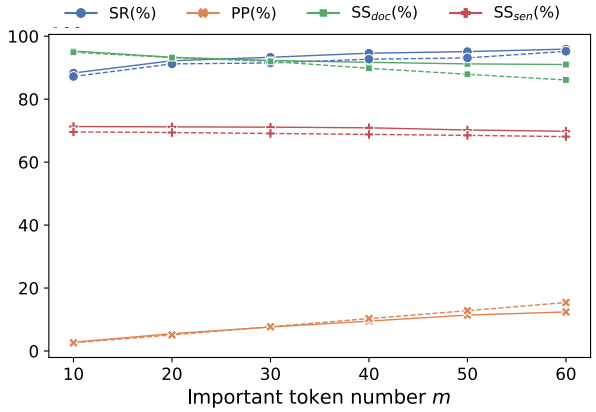

Fig. 5. Performance comparison between PRADA and TS$_{rep}$ with different numbers of important tokens on MS-MARCO-Doc. Dotted lines denote TS$_{rep}$; solid lines denote PRADA.

Table 9. Adversarial samples generated by TS$_{rep}$ and PRADA on the MS-MARCO-Doc dataset. The perturbed words are marked as blue and red/magenta in the original document and adversarial example, respectively.

Method	Query: "government does do"	Rank Position
Original	… what kind of government does japan have today? answered by the wiki answers community answers. com is making the world better one answer at a time. japan is a constitutional monarchy with a parliamentary government. the constitution. it awards the vote to all men and women age 20 and older. was this answer useful? what kind of government did japan have? japan has the type of government like canada …	60
TS$_{rep}$	… what kind of government does japan have today? answered by the wiki answers community answers. com is making the world better one answer at a time. japan is a does do government does do government does do government does do government does do government does do government does do government does do government does do government does do government does do government does do government does do government does do government does do government does do like canada …	38
PRADA	… what kind of government does japan have currently? answer by the wiki answers community answers. com is making the world better one answer at a time. japan is a constitution monarchy with a parliamentary government. the constitution. he awards the vote to all men and women age 20 and older. was this answer helpful? what kind of government did japan has? japan has the types of government like canadian …	35

PRADA is more semantically consistent with the original document by human judges, while the rank position given by the target model is higher (i.e., 35 vs. 38). It indicates that while obvious query term attack TS$_{rep}$ still has little effect on the rank promotion of documents in some cases,

our PRADA can generate human-imperceptible perturbations to the document and promote its rankings to a greater extent. (2) The adversarial document generated by TS_{rep} has a wider range of obvious replacements with query terms, making them distinguishable from the original document and less fluent. Word-level synonyms seem more reasonable for guaranteeing fluency and semantic preservation in adversarial samples than the query terms.

7 CONCLUSION AND FUTURE WORK

In this paper, we introduced a challenging WSRA task against NRMs, which aims to promote a target document in rankings by adding adversarial perturbations to its text. We focused on the practical decision-based black-box attack setting and developed a novel method PRADA based on the PRF idea to generate the adversarial examples for effective attack. Empirical results show that PRADA achieves a high success rate with small indiscernible perturbations. Besides, PRADA can evade the detection of the anti-spamming method easily. The findings show that NRMs are very vulnerable to adversarial attacks which promotes the document in rankings with human-imperceptible perturbations.

One limitation of our PRADA is that the attack may fail the short documents or low-ranked documents. Meanwhile, the computational cost is relatively high since it requires to generate the adversarial document for any test document in the inference time.

In future work, we aim to pursue stronger black-box attacks against NRMs. Furthermore, exploring how to combine the adversarial objectives with original learning objectives in the training phrase will also be a potential direction to mitigate the computational cost. It is also valuable to attack a real-world search engine using PRADA to demonstrate its practical applicability. Since the NRMs are very vulnerable to adversarial attacks, it is critical to develop the corresponding defense methods to enhance the robustness of NRMs before they are widely deployed to real-world search engines. A straightforward idea is to leverage the prevalent adversarial training [14] to conduct the defense. In adversarial training, adversarial examples (e.g., generated by PRADA) are found during training and used to augment the training set. Besides, since the empirical defense offers no theoretical guarantee on the models' robustness and may eventually be broken by other sophisticated adversarial attacks, certified defense [**?**] can also be developed to give rigorous and provable certified robustness. The defense method needs to be elaborated to consider the relationship between the attacked document and the query as well as other candidate documents for IR. We hope our study provides useful clues for future research on adversarial ranking defense and helps to develop robust real-world search engines.

REPRODUCIBILITY

To facilitate the reproducibility of the results reported in this paper, we share the following resources: code for PRADA. See https://github.com/wuchen95/PRADA.

ACKNOWLEDGMENTS

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 62006218 and 61902381, the Youth Innovation Promotion Association CAS under Grants No. 20144310 and 2021100, the Young Elite Scientist Sponsorship Program by CAST under Grants No. YESS20200121, and the Lenovo-CAS Joint Lab Youth Scientist Project. This work was also (partially) funded by the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research, https://hybrid-intelligence-centre.nl. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. arXiv preprint arXiv:1804.07998 (2018).

[2] Anish Athalye and Nicholas Carlini. 2018. On the robustness of the CVPR 2018 white-box adversarial example defenses. arXiv preprint arXiv:1804.03286 (2018).

[3] Andras A Benczur, Karoly Csalogany, Tamas Sarlos, and Mate Uher. 2005. SpamRank–Fully automatic link spam detection. In AIRWeb.

[4] Carlos Castillo and Brian D Davison. 2011. Adversarial web search. Found. Trends Inf. Retr. 4, 5 (2011), 377–486.

[5] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. arXiv preprint arXiv:1803.11175 (2018).

[6] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. 2020. Hopskipjumpattack: A query-efficient decision-based attack. In SP. IEEE, 1277–1294.

[7] Mingyang Chen, Junda Lu, Yi Wang, Jianbin Qin, and Wei Wang. 2021. DAIR: A query-efficient decision-based attack on image retrieval Systems. In SIGIR.

[8] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the TREC 2019 deep learning track. arXiv preprint arXiv:2003.07820 (2020).

[9] Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for IR with contextual neural language modeling. In SIGIR. 985–988.

[10] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. 2017. Neural ranking models with weak supervision. In SIGIR. 65–74.

[11] 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).

[12] Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. EPM 33, 3 (1973).

[13] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In SPW. IEEE, 50–56.

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014).

[15] Gregory Goren, Oren Kurland, Moshe Tennenholtz, and Fiana Raiber. 2018. Ranking robustness under adversarial document manipulations. In SIGIR.

[16] Gregory Goren, Oren Kurland, Moshe Tennenholtz, and Fiana Raiber. 2020. Ranking-Incentivized Quality Preserving Content Modification. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 259–268.

[17] Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. 2020. Speaker-aware BERT for multi-turn response selection in retrieval-based chatbots. In CIKM. 2041–2044.

[18] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In CIKM. 55–64.

[19] Zoltan Gyongyi and Hector Garcia-Molina. 2005. Web spam taxonomy. In AIRWeb.

[20] Zoltan Gyongyi, Hector Garcia-Molina, and Jan Pedersen. 2004. Combating web spam with TrustRank. In VLDB.

[21] Peter Izsak, Fiana Raiber, Oren Kurland, and Moshe Tennenholtz. 2014. The search duel: a response to a strong ranker. In Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. 919–922.

[22] Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. arXiv preprint arXiv:1707.07328 (2017).

[23] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In AAAI, Vol. 34. 8018–8025.

[24] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. TACL 8 (2020), 64–77.

[25] Hang Li. 2014. Learning to rank for information retrieval and natural language processing. Synthesis Lectures on Human Language Technologies 7, 3 (2014), 1–121.

[26] Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. arXiv preprint arXiv:1812.05271 (2018).

[27] Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. arXiv preprint arXiv:2004.09984 (2020).

[28] Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2017. Deep text classification can be fooled. arXiv preprint arXiv:1704.08006 (2017).

[29] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2021. Pretrained transformers for text ranking: BERT and beyond. Synthesis Lectures on Human Language Technologies 14, 4 (2021), 1–325.

[30] Tie-Yan Liu. 2011. Learning to Rank for Information Retrieval. Springer Science & Business Media.
[31] Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Xiang Ji, and Xueqi Cheng. 2021. PROP: Pre-training with representative words prediction for ad-hoc retrieval. In WSDM. 283–291.
[32] Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Yingyan Li, and Xueqi Cheng. 2021. B-PROP: Bootstrapped pre-training with representative words prediction for ad-hoc retrieval. arXiv preprint arXiv:2104.09791 (2021).
[33] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017).
[34] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In EMNLP. 404–411.
[35] Pasquale Minervini and Sebastian Riedel. 2018. Adversarially regularising neural NLI models to integrate logical background knowledge. arXiv preprint arXiv:1808.08609 (2018).
[36] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to match using local and distributed representations of text for web search. In WWW.
[37] Nikola Mrkšić, Diarmuid O Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. arXiv preprint arXiv:1603.00892 (2016).
[38] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In CoCo@ NIPS.
[39] Alexandros Ntoulas, Marc Najork, Mark Manasse, and Dennis Fetterly. 2006. Detecting spam web pages through content analysis. In WWW. 83–92.
[40] Kezban Dilek Onal, Ye Zhang, Ismail Sengor Altingovde, Md Mustafizur Rahman, Pinar Karagoz, Alex Braylan, Brandon Dang, Heng-Lu Chang, Henna Kim, Quinten McNamara, Aaron Angert, Edward Banner, Vivek Khetan, Tyler McDonnell, An Thanh Nguyen, Dan Xu, Byron C. Wallace, Maarten de Rijke, and Matthew Lease. 2018. Neural information retrieval: At the end of the early years. Information Retrieval Journal 21, 2–3 (June 2018), 111–182.
[41] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277 (2016).
[42] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In ASIA. 506–519.
[43] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In EMNLP. 1532–1543.
[44] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018).
[45] Jakub Piskorski, Marcin Sydow, and Dawid Weiss. 2008. Exploring linguistic features for web spam detection: a preliminary study. In AIRWeb. 25–28.
[46] Jay M Ponte and W Bruce Croft. 1998. A language modeling approach to information retrieval. In SIGIR. 275–281.
[47] Nisarg Raval and Manisha Verma. 2020. One word at a time: Adversarial attacks on retrieval models. arXiv preprint arXiv:2008.02197 (2020).
[48] Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In SIGIR'94. Springer, 232–241.
[49] Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. Commun. ACM 18, 11 (1975), 613–620.
[50] Asim Shahzad, Nazri Mohd Nawi, Muhammad Zubair Rehman, and Abdullah Khan. 2021. An improved framework for content-and link-based web-spam detection: A combined approach. Complexity 2021 (2021).
[51] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In CVPR. 1–9.
[52] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013).
[53] Clifford Tatum. 2005. Deconstructing Google bombs: A breach of symbolic power or just a goofy prank? First Monday 10, 10 (2005).
[54] Chao Wei, Yiqun Liu, Min Zhang, Shaoping Ma, Liyun Ru, and Kuo Zhang. 2012. Fighting against web spam: A novel propagation method based on click-through data. In SIGIR. 395–404.
[55] Jincheng Xu and Qingfeng Du. 2020. TextTricker: Loss-based and gradient-based adversarial attacks on text classification models. Engineering Applications of Artificial Intelligence 92 (2020), 103641.
[56] Ying Xu, Xu Zhong, Antonio Jimeno Yepes, and Jey Han Lau. 2021. Grey-box Adversarial Attack And Defence For Sentiment Classification. arXiv preprint arXiv:2103.11576 (2021).
[57] Erkun Yang, Tongliang Liu, Cheng Deng, and Dacheng Tao. 2018. Adversarial examples for hamming space search. IEEE transactions on cybernetics 50, 4 (2018).
[58] Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible ranking baselines using Lucene. JDIQ 10, 4 (2018), 1–20.

[59] Bin Zhou and Jian Pei. 2009. OSD: An online web spam detection system. In KDD, Vol. 9.
[60] Mo Zhou, Zhenxing Niu, Le Wang, Qilin Zhang, and Gang Hua. 2020. Adversarial ranking attack and defense. In ECCV. Springer, 781–799.