Tanmay_Dissertation (2).pdf

Indian Statistical Institute

Document Details

Submission ID
trn:oid:::3618:100555273

Submission Date
Jun 12, 2025, 5:20 PM GMT+5:30

Download Date
Jun 12, 2025, 5:46 PM GMT+5:30

File Name
Tanmay_Dissertation (2).pdf

File Size
903.1 KB

49 Pages

9,406 Words

49,498 Characters

12% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Custom Section Exclusions

{titlesCount} Section Titles, {keywordsCount} Keywords

Section title	No. of Section Starters	Section Starters
"isi"	2	Indian statistical institute, kolkata

Match Groups

- **77** Not Cited or Quoted 11%
  Matches with neither in-text citation nor quotation marks
- **9** Missing Quotations 1%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

- 10% Internet sources
- 10% Publications
- 0% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **77** Not Cited or Quoted 11%
  Matches with neither in-text citation nor quotation marks
- **9** Missing Quotations 1%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

10% Internet sources
10% Publications
0% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	p.pdfhall.com 1%
2	Internet	ip.ios.semcs.net <1%
3	Internet	arxiv.org <1%
4	Internet	repository.ubn.ru.nl <1%
5	Publication	Shilin Qiu, Qihe Liu, Shijie Zhou, Wen Huang. "Adversarial Attack and Defense Tec... <1%
6	Internet	dspace.isical.ac.in:8080 <1%
7	Internet	export.arxiv.org <1%
8	Internet	library.isical.ac.in:8080 <1%
9	Internet	www.dei.unipd.it <1%
10	Internet	deepai.org <1%

11 Publication

Jaime, Koh Ming Wen. "K-Periodic Scheduling for Throughput-Buffering Trade-Off... <1%

12 Internet

d-nb.info <1%

13 Publication

Haibin Zheng, Jinyin Chen, Wenchang Shangguan, Zhaoyan Ming, Xing Yang, Zhij... <1%

14 Internet

www.arxiv-vanity.com <1%

15 Publication

"Advances in Information Retrieval", Springer Science and Business Media LLC, 20... <1%

16 Publication

Chen Wu, Ruqing Zhang, Jiafeng Guo, Maarten De Rijke, Yixing Fan, Xueqi Cheng. ... <1%

17 Publication

Negar Arabzadeh, Maryam Khodabakhsh, Ebrahim Bagheri. "BERT-QPP: Contextu... <1%

18 Internet

aoip.osa.org <1%

19 Internet

pure.tue.nl <1%

20 Internet

wiredspace.wits.ac.za <1%

21 Publication

Moraffah, Raha. "Responsible Machine Learning: Security, Robustness, and Causa... <1%

22 Internet

ceur-ws.org <1%

23 Internet

web.archive.org <1%

24 Internet

fatcat.wiki <1%

25 Publication
"Advances in Information Retrieval", Springer Science and Business Media LLC, 20... <1%

26 Publication
"Advances in Information Retrieval", Springer Science and Business Media LLC, 20... <1%

27 Internet
m.moam.info <1%

28 Internet
dspace.christcollegeijk.edu.in:8080 <1%

29 Internet
github.com <1%

30 Internet
ir.lib.nycu.edu.tw <1%

31 Publication
"Advances in Information Retrieval", Springer Science and Business Media LLC, 20... <1%

32 Internet
10wvvw.easychair.org <1%

33 Publication
Abualeenein, Abdelrahman. "Performance Study of Darrieus Turbine Using Nume... <1%

34 Publication
Chen Wu, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, Xueqi Cheng. ... <1%

35 Internet
theses.hal.science <1%

36 Internet
www.scaler.com <1%

37 Publication
Heyuan Shi, Binqi Zeng, Yu Zhan, Rongkai Liu, Yulin Yang, Li Chen, Chao Hu, Ying ... <1%

38 Publication
Shengyao Zhuang, Xinyu Mao, Guido Zuccon. "Robustness of Neural Rankers to T... <1%

39 Internet
etheses.whiterose.ac.uk <1%

40 Internet
hdl.handle.net <1%

41 Internet
link.springer.com <1%

42 Internet
opus.lib.uts.edu.au <1%

43 Publication
Emrah Budur, Rıza Özçelik, Dilara Soylu, Omar Khattab, Tunga Güngör, Christoph... <1%

44 Publication
Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Jimmy Lin. "MS MAR... <1%

45 Publication
Reddy, Siddhant. "Unsupervised Transfer Learning with Autoencoders", Rocheste... <1%

46 Publication
Maryam Khodabakhsh, Ebrahim Bagheri. "Learning to Rank and Predict: Multi-Ta... <1%

47 Publication
Negar Arabzadeh, Radin Hamidi Rad, Maryam Khodabakhsh, Ebrahim Bagheri. "... <1%

48 Publication
Samoska, Nevena. "Evaluation and performance prediction of multimodal biomet... <1%

49 Publication
Sebastian Hofstätter, Aldo Lipani, Markus Zlabinger, Allan Hanbury. "Learning to ... <1%

50 Internet
christoph-conrads.name <1%

51 Publication
Liu, Han. "Towards Secure and Privacy-Preserving Machine Learning Systems.", W... <1%

52 Publication
"Advances in Information Retrieval", Springer Science and Business Media LLC, 20... <1%

53 Publication

István Megyeri. "Applications of Adversarial Robustness Analysis in Machine Lear... <1%

Word Level Attack for Text Ranking

Dissertation submitted in partial fulfilment of the requirements
for the award of the degree of

Master of Technology in Computer Science

by

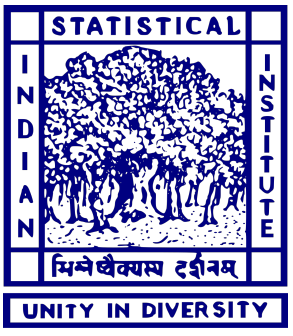
Tanmay Karmakar

(Roll No. CS2333)

Under the Supervision of

Dr. Debapriyo Majumdar

Computer Vision and Pattern Recognition Unit



INDIAN STATISTICAL INSTITUTE, KOLKATA

Kolkata - 700108, India

June, 2025

Certificate

I attest that **Tanmay Karmakar**'s thesis, titled "**Word Level Attack for Text Ranking**," which was prepared under my guidance, meets all necessary standards in scope and quality. In my professional judgment, it is entirely suitable as a dissertation for the **Master of Technology in Computer Science** degree awarded by the Indian Statistical Institute.

.....

Debapriyo Majumdar
Assistant Professor
Computer Vision and Pattern Recognition Unit,
Indian Statistical Institute Kolkata.

Declaration of Authorship

I, Tanmay Karmakar, declare that this thesis titled, **Word Level Attack for Text Ranking** and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a Master's degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

. . . . . .

Tanmay Karmakar
Roll No.: CS2333
Date: June 2025
ISI Kolkata

Acknowledgements

I would like to express my sincere gratitude to my research guides, **Dr. Debapriyo Majumdar**, for their invaluable guidance and support throughout this research. Their expertise and insights have been instrumental in shaping this thesis.

Special thanks to **Sourav Saha**, research scholar pursuing his Ph.D, who guided me through various aspects of this research. His patience and knowledge were crucial in overcoming numerous challenges.

I would also like to thank my classmate and colleague, **Surjyanee Halder**, for her collaboration. Her support and cooperation have made this journey more manageable and enjoyable.

Thank you all for your patient support and belief in my work.

Tanmay Karmakar

INDIAN STATISTICAL INSTITUTE

Abstract

Word Level Attack for Text Ranking

Neural Ranking Models (NRMs) have become state-of-the-art in information retrieval, demonstrating remarkable effectiveness across various search and ranking tasks. However, their increasing deployment in real-world systems raises critical concerns about their robustness and susceptibility to adversarial attacks. This project investigates the fragility of modern NRMs by proposing and evaluating a document perturbation method based on targeted, single-word perturbation. Our approach strategically identifies an influential word depending on the query to be substituted or added in the document. We have done experiments on benchmark datasets to assess the impact of these minimal perturbations on ranking performance. Our findings reveal that even a single carefully chosen word addition or substitution can significantly change the ranking score of the targeted document providing insight into the NRMs.

Contents

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Information Retrieval (IR) is the science and art of finding relevant information from large collections of unstructured data to satisfy a user's information need. At its core, IR is concerned with representing, storing, organizing, and providing access to information resources. The process typically begins when a user expresses their need as a **query**. In response, the system searches its repository to find a set of potentially relevant documents. The most critical step in modern IR is **ranking**, where a sophisticated model scores and sorts the retrieved documents to present the most relevant documents are at the top. The effectiveness of an IR system is thus measured by its ability to retrieve relevant documents while filtering out irrelevant ones. These systems are integral to our daily digital interactions, powering everything from web search engines like Google and academic databases such as IEEE Xplore, to e-commerce product searches on platforms like Amazon and even the recommendation systems that suggest content on Netflix and Spotify. This project is situated within the sub-field of text ranking, focusing specifically on the robustness of the neural models that power these sophisticated systems.

1.1 Neural Ranking Models in IR

Ranking of documents according to the relevance to a query is central to Information Retrieval. Historically, this was predominantly addressed by **statistical**

ranking models, such as BM25 or language modeling approaches, which rely on term frequencies, document lengths, and other statistical properties of text. While these methods laid a strong foundation and are still effective in many scenarios, the landscape of IR has been significantly reshaped by the advent of deep learning. In recent years, **Neural Ranking Models (NRMs)** have emerged as the dominant paradigm, leveraging complex neural architectures, often based on Transformers like BERT, to learn intricate patterns of relevance between queries and documents. These models, trained on large-scale datasets, have demonstrated state-of-the-art performance by capturing deeper semantic relationships that elude traditional statistical methods. Consequently, NRMs are now widely adopted in both academic research and commercial search engines, forming the backbone of modern information access systems. This dissertation focuses on the robustness of these advanced NRMs, a critical aspect given their widespread deployment.

1.2 Adversarial Attacks on Neural Ranking Models

Despite the impressive performance of Neural Ranking Models (NRMs), their vulnerability to adversarial attacks poses a serious concern for NRMs' deployment in real-world applications. An adversarial attack involves making small, often imperceptible, modifications to a document's text with the intent of influencing the model's output — for instance, by promoting a non-relevant document or demoting a relevant one in the ranking.

These attacks can be broadly categorized based on the level at which text is modified:

- **Character-level attacks:** These attacks operate at the finest granularity. They involve modifying individual characters within words — such as inserting, deleting, swapping, or replacing letters (e.g., changing 'support' to 'sup0rt'). Though the changes may look like typos, they can fool models that rely heavily on sub-word tokenization. Such attacks are often imperceptible to humans but may affect token embeddings significantly.
- **Word-level attacks:** These are among the most practical and studied forms of adversarial perturbation, in this type of adversarial attack, some words

are chosen and replaced of some words are inserted into the document. This can significantly affect the perceived semantic relevance of a document without making the modification obvious. Since our work focuses on minimal document changes, we concentrate on this attack type and its effectiveness in ranking manipulation.

- **Sentence-level attacks:** At the coarsest level, adversaries manipulate entire sentences. This can involve paraphrasing, sentence reordering, insertion, or deletion. For example, rewording "The phone battery lasts long" to "Battery life is above average" changes the phrasing without altering meaning. Such changes can affect the contextual interpretation of documents, though generating fluent and plausible sentence-level attacks is more computationally demanding.

Understanding the behavior of NRMs under each of these attack categories is essential for designing more robust and interpretable retrieval systems. This dissertation specifically investigates word-level substitutions as a minimal yet effective adversarial strategy.

1.2.1 Word-level Attacks on NRMs

Word-level attacks are widely adopted due to their balance between effectiveness and semantic coherence. These attacks typically preserve grammatical structure and readability while significantly altering model behavior. Two primary types of word-level attacks are described below:

- **Word Substitution:** This involves replacing a word in the document with another that changes the model's interpretation of the content. The strategy comprises two main steps:
  - ***Choosing the word to substitute*** — Identify a word that has significant influence on the document's relevance with respect to the given query. This is typically done using relevance attribution methods such as saliency scores, attention weights, or gradient-based signals.

  - ***Selecting the substitute word*** — Replace the chosen word with a suitable alternative, which can be a synonym, antonym, or a term highly relevant to the query context. The replacement should ideally preserve fluency while inducing a shift in the ranking score.

- **Word Addition:** This attack adds new terms to the document to influence the ranking outcome. The strategy includes:

  - ***Choosing the word to add*** — Select a word that is contextually or semantically aligned with the query or that the model may consider as a relevance signal. This could be a frequently co-occurring keyword or a term with high impact in similar documents.
  - ***Deciding the insertion point*** — Determine where to insert the word within the document. Common strategies include placing the word at the beginning, end, or near existing query-related content to maximize influence while maintaining sentence coherence.

This dissertation focuses on both word substitution and word addition as minimal yet effective methods for generating adversarial perturbations against NRMs.

1.2.2 Black-box vs. White-box Models

We have talked about different types of adversarial attacks depending on the perturbation strategy on the document. The adversarial attack can also be categorized based on the available information of the Neural ranker model that we are attacking. In a ***black-box*** setting, the attacker has no internal knowledge of the model parameters or gradients. Only the input-output behavior (e.g., rankings) is observable, making attacks dependent on query probing or surrogate model training.

Conversely, in a ***white-box*** setting, full access to the NRM's architecture, weights, and gradients is available. This enables gradient-based attacks that can pinpoint sensitive positions in the input text, such as tokens with high influence on the ranking score.

Black-box attacks are more realistic in practice but often less effective, while white-box attacks, though powerful, assume access that may not be feasible in deployed

systems. This work primarily assumes a white-box setting to demonstrate the vulnerability of NRMs under ideal adversarial conditions.

1.3 Problem Statement

The increasing adoption of Neural Ranking Models (NRMs) in modern information retrieval systems raises critical questions about their robustness in adversarial settings. These models, despite their impressive performance, may be susceptible to minor, targeted perturbations in input documents.

This dissertation aims to systematically analyze the robustness of NRMs against word-level adversarial perturbations. Specifically, we focus on minimal manipulations — such as the substitution or addition of a single word — and investigate how these affect the ranking behavior of state-of-the-art neural rankers. The goal is to evaluate the extent to which these models can be manipulated by semantically subtle edits and to identify potential vulnerabilities in their decision-making processes.

1.3.1 Task Description

In this sub-section we are going to discuss more about what we are trying to achieve in this Dissertation. Similar to PRADA [1], given a query q and a document d from corpus $\mathcal{C}$, a Neural Ranker Model f aims to predict the relevance score $f(q, d)$. Typically, for a query q we first retrieve a candidate set of documents $\mathcal{D} = \{d_1, d_2, \cdots, d_n\}$, then the ranking model f assign relevance score $f(q, d_i)$ for each of the documents in $\mathcal{D}$ which are then used to rank the candidate documents. For example, the model f outputs $L = [d_N, d_{N-1}, \cdots, d_1]$ if it determines $f(q, d_N) > f(q, d_{N-1}) > \cdots > f(q, d_1)$.

Based on these settings, the objective is to promote the rank of a selected document d_k, where $k \neq N$, from the ranked list L produced by the model f, through a minimal word-level perturbation applied to d_k guided by the query q. Formally, we say, d_k^{adv} is a successful perturbation of the document d_k if it satisfies the following,

$$Rank_L(d_k^{adv}) < Rank_L(d_k) \tag{1.1}$$

where d_k^{adv} can be described as d_k+p where p is the query guided perturbation applied on d_k. In this case we call d_k^{adv} a valid **adversarial example** of the document d_k, and p is called a valid perturbation.

In this dissertation, we talked about some of the previous work done related to our task in Chapter 2, then we proposed a method to find a valid one-word perturbation to a targeted document in Chapter 3. The experimental setup, the results, and the key inferences are shown in Chapter 4. Followed by the conclusion and some aspects of future work in Chapter 5.

Chapter 2

Related Work

This chapter surveys the landscape of research on adversarial attacks against neural ranking models (NRMs). The vulnerability of Deep Neural Network models to adversarial attacks were first examined and explored in the image domain [2]. Later adapted in the Natural Language Processing domain, this approach of adversarial attack spurred a significant amount of research that has been done to improve the robustness of NRMs. Similar to [3], which manipulates only one pixel of an image to fool neural model our approach focuses to perturbing/adding only one word into some document to fool NRMs. The next sections cover some work on adversarial attacks on the language domain.

2.1 Character Level Attack

Character-level attacks represent the most fine-grained form of textual perturbation, involving subtle modifications like the insertion, deletion, swapping, or replacement of individual characters within words [4]. For instance, techniques such as HotFlip [5] have been used to identify and alter characters that have a high impact on

the model's output based on gradient information. While these modifications can effectively fool models that are sensitive to sub-word tokenization or mimic real-world typos, they often risk creating non-sensical words or being easily flagged by spelling correctors [6]. Due to their tendency to disrupt lexical integrity, they are often considered less practical for semantic ranking tasks compared to word-level attacks, which better preserve document readability.

2.2 Word Level Attack

A significant focus of recent work is on developing practical attack methods that mirror real-world scenarios. The PRADA framework, proposed by Wu et al. [1], operates in a challenging *decision-based black-box* setting, where an attacker has no access to model parameters or gradients and can only observe the final rank of documents. To overcome this lack of information, PRADA[1] trains a separate surrogate ranking model. It uses Pseudo-Relevance Feedback (PRF) by querying the target model and treating the top-ranked results as positive examples to train this surrogate. The gradients from this surrogate model are then used to guide a Word Substitution Ranking Attack (WSRA), which effectively promotes a target document's rank by making small, imperceptible changes to its text. Their results show this method can successfully fool well-trained NRMs with a high success rate.

In contrast, Wang et al. [7] investigate the fragility of BERT-based rankers from a *white-box* perspective, where full access to the model's gradients is assumed. Their gradient-based approach demonstrates that adding or replacing as few as five tokens can cause dramatic shifts in a document's rank. Their study uncovers two critical vulnerabilities: first, BERT rankers are disproportionately sensitive to perturbations

at the beginning of a document's text; and second, a small, recurring set of adversarial tokens can be effective across many different queries, exposing potential biases in either the pre-training or fine-tuning datasets.

Complementing research on attack execution, the work of Ivgi and Berant [8] explores how adversarial examples can be used to fortify models. They systematically compare **offline augmentation**, where a static set of adversarial examples is generated once for retraining, against **online augmentation**, where adversarial examples are generated dynamically at every training step. Their findings show that online augmentation leads to the most significant gains in model robustness, as the attacks adapt to the model as it learns. Furthermore, they reveal that simple random sampling of perturbations can be more effective and significantly faster than complex, search-based offline methods. Their work also contributes an improved search algorithm, Best-First search over a Factorized graph (BFF), designed to find discrete adversarial examples more effectively than previous greedy approaches.

Chapter 3

Proposed Methods

This chapter presents our proposed method for generating adversarial perturbations against Neural Ranking Models (NRMs) using a single-word manipulation strategy. The core idea is to identify the most influential word in the user query — referred to as the *query center* — and insert it into the target document at a strategically chosen location. This minimal addition is designed to subtly boost the semantics of the document with respect to the query, thereby altering its position in the ranked list. By focusing on query-aware word insertion rather than arbitrary modifications, our method ensures high attack success rates. The simplicity and effectiveness of this one-word attack highlight the vulnerability of NRMs to small but targeted perturbations.

3.1 Attacking Method: One-Word Manipulation

The core attack strategy proposed in this work involves a minimal perturbation through the addition of a single, carefully chosen word to a target document. The

process is outlined as follows:

1. Let q be a given query. Using this query, the top 100 documents are first retrieved via the BM25 algorithm. These documents are then reranked using a Neural Ranking Model (NRM), resulting in a new relevance-based order.

2. From the reranked list, a single document d is selected as the target for adversarial manipulation.

3. The most informative or central word from the query, referred to as the *query center* (q_{center}), is identified. We will describe the query center later in detail 3.1.1.

4. The query center word is then inserted into document d at a suitable location, yielding the perturbed document $d' = d + q_{\text{center}}$.

The insertion location is chosen to maximize the influence of the added word on the NRM's scoring function while preserving the document's coherence. This one-word manipulation serves as a low-cost yet effective adversarial strategy.

3.1.1 Query Center

To execute a targeted word-level attack, it is crucial to identify the most semantically central term in the user query — referred to as the *query center*. This word serves as the basis for generating effective perturbations in the document. The process for computing the query center is outlined below:

1. Let the input query be $q = (q_1, q_2, \ldots, q_k)$, where each q_i represents an individual query term.

2. Each term q_i is mapped to a high-dimensional vector $\mathbf{v}(q_i)$ using a *counter-fitted word embedding space* [9], in which semantically similar words are positioned close together. These embeddings enhance synonym sensitivity by enforcing geometric constraints between word vectors.

3. Compute the centroid of the query term vectors:

$$\mathbf{v}(q) = \frac{1}{k} \sum_{i=1}^{k} \mathbf{v}(q_i)$$

   This centroid vector $\mathbf{v}(q)$ captures the average semantic direction of the query.

4. Search the embedding space to identify the word q_{center} whose vector is closest (typically under cosine similarity) to the centroid vector $\mathbf{v}(q)$.

5. The retrieved word q_{center} is selected as the *query center*.

This method ensures that the chosen query center reflects the overall semantic intent of the query and is thus a strong candidate for insertion during the adversarial attack. The following algorithm summaries how to find the query center.

Algorithm 1 Find query center

Input: Query $q = (q_1, q_2, \ldots, q_k)$, counter-fitted word embedding space

Output: Query center word q_{center}

1: **for** each term $q_i \in q$ **do**

2: Compute word vector $\mathbf{v}(q_i)$ using the counter-fitted embedding space

3: **end for**

4: Compute centroid vector:

$$\mathbf{v}(q) = \frac{1}{k} \sum_{i=1}^{k} \mathbf{v}(q_i)$$

5: Find the word q_{center} in the embedding space such that $\mathbf{v}(q_{\text{center}})$ is closest to $\mathbf{v}(q)$ (e.g., under cosine similarity)

6: **return** q_{center}

3.2 Perturbation Using the Query Center

In this section, we discussed how to use the query center q_{center} to perturb a retrieved document so that its rank decreases. Let q be a user query and L be the ranked list of documents produced by a neural ranking model f. We denote d_k as the targeted document selected from L, where $k \neq N$ and N is the index of the top-ranked document. Our goal is to adversarially perturb d_k in a minimal yet effective way using the computed query center q_{center}, such that its rank is improved within the list L.

We propose three distinct strategies for perturbing the target document: `one_word_start`, `one_word_sim`, and a gradient-based approach, `one_word_grad` with an improved variant `one_word_best_grad`. for injecting or substituting q_{center}

into the target document d_k. Each method aims to subtly manipulate the model's ranking score for d_k with minimal alteration to its content and fluency.

3.2.1 Method 1: one_word_start

The simplest black-box model approach involves inserting q_{center} at the beginning of the document d_k. As shown in [7], this method also assumes that terms introduced early in the text can have a stronger influence on the model's scoring function due to positional sensitivity.

Algorithm 2 one_word_start

Input: Target document d_k, query q, ranking model f

Output: Final document d^*

Find q_{center} using Algorithm 1

Insert q_{center} at the beginning of d_k

Construct perturbed document:

$$d'_k = q_{\text{center}} \,\|\, d_k$$

if $\text{Rank}_L(d'_k) < \text{Rank}_L(d_k)$ **then**

  return d'_k

else

  return d_k

end if

3.3 Method 2: one_word_sim

23 The `one_word_sim` strategy perturbs the document by replacing a word in the target document with the query center q_{center}. Unlike insertion-based techniques, this approach preserves the document length while introducing query-relevant semantics with minimal disruption to fluency.

We first compute the query center q_{center} from the query q using the method described earlier. Then, for each token t_i in the document d_k, we find the similarity between t_i and the query center q_{center}.

Next, we identify the word t^* in the document that is most similar to q_{center}, ensuring $t^* \neq q_{\text{center}}$ to avoid self-substitution. This word is considered the optimal candidate for replacement due to its semantic closeness to the query center.

Finally, we substitute t^* with q_{center}, resulting in the perturbed document d'_k. This substitution is minimal yet semantically aligned with the query, potentially influencing the model's ranking without compromising readability or naturalness.

Algorithm 3 one_word_sim

Input: Query q, target document d_k, ranking model f

Output: Perturbed document d'_k

Compute query center q_{center} from q using Algorithm 1
Initialize: min_distance $\leftarrow \infty$, $t^* \leftarrow$ None
for each token t_i in d_k **do**
  if $t_i \neq q_{\text{center}}$ **then**
    Compute similarity distance between t_i and q_{center}
    if distance $<$ min_distance **then**
      min_distance $\leftarrow$ distance, $t^* \leftarrow t_i$
    end if
  end if
end for
Replace t^* in d_k with q_{center} to form temporary document d'_k
if $\text{Rank}_L(d'_k) < \text{Rank}_L(d_k)$ **then**
  return d'_k
else
  return d_k
end if

3.3.1 Method 3A: one_word_grad

As the name suggests, we need the model's gradient values for this type of attack makes the model a white-box model. This is the most complex method that we proposed. Similar to the other settings, let q be the query and $L = [d_N, d_{N-1}, \cdots, d_1]$ be the ranked list outputted by the NRM f for the query q.

We adopt the gradient-based ranking attack framework proposed in PRADA [1] to guide the insertion of the query center q_{center} into the target document. In this approach, the most influential positions in the document are identified based on their gradient contributions to a hinge loss function, allowing for effective white-box perturbation.

PRADA [1] defines the pairwise hinge loss used for relevance ranking as:

$$\mathcal{L}_{\text{rank}}(q, d_k + p; L) = \sum_{d' \in L \setminus \{d_k\}} \max(0, \beta - f(q, d_k + p) + f(q, d')) \tag{3.1}$$

where f is the neural ranker, L is the original ranked list, d_k is the target document, and β is a margin hyperparameter.

To identify impactful position in the text, the importance of each token t_i in the document is computed using the gradient of the hinge loss with respect to its embedding $\mathbf{e}_{t_i}$:

$$I(t_i) = \left\| \frac{\partial \mathcal{L}_{\text{rank}}}{\partial \mathbf{e}_{t_i}} \right\|_2^2 \tag{3.2}$$

Tokens with higher $I(t_i)$ are considered more influential in affecting the rank of d_k.

In our proposed method, we use this importance score to find the most sensitive position in the document. We then insert the query center word q_{center} at that position to construct the perturbed document d'_k. This allows us to perform a minimal, query-aware, and highly targeted adversarial attack in the white-box setting.

Algorithm 4 one_word_grad

Input: Query q, target document d_k, ranking model f, query center word q_{center}

Output: Perturbed document d'_k

Compute hinge loss $\mathcal{L}_{\text{rank}}(q, d_k; L)$ Equation 3.1

for each token t_i in d_k **do**

    Compute importance score $I(t_i)$ Equation 3.2

end for

Identify position $i^* = \arg\max_i I(t_i)$

Insert q_{center} at position i^* in d_k to construct perturbed document d'_k

if $\text{Rank}_L(d'_k) < \text{Rank}_L(d_k)$ **then**

    return d'_k

else

    return d_k

end if

3.3.2 Method 3B: one_word_best_grad

In the **one_word_grad** algorithm, we have used the position of the document with the maximum gradient of the hinge loss (Equation 3.1). Although that gives a notion of the most important position in the document, that might not be the optimal position to increase the relevance score by inserting the query center. But if we try to find the optimal position by testing all the positions of the document, the required time will be proportional to the length of the document, which can be too much. So, we have used the importance score (Equation 3.2) to select k candidate positions in the document, where k is a constant (we have used $k = 20$).

Selecting Candidate Positions: The selection process of the k candidate positions in the document is rather simple. We just select the top k positions with the

highest importance score. Then we test each of the candidate positions by inserting the query center and computing the relevance score using the model. Finally, the position that increases the relevance score the most is selected for perturbation.

Algorithm 5 one_word_best_grad

Input: Query q, target document d_k, ranking model f, query center word q_{center}, number of candidates k

Output: Perturbed document d_k^*

Compute hinge loss $\mathcal{L}_{\text{rank}}(q, d_k; L)$ Equation 3.1
for each token t_i in d_k **do**
    Compute importance score $I(t_i)$ Equation 3.2
end for
Select top-k positions $\{i_1, i_2, \ldots, i_k\}$ with highest importance scores
Initialize best score $s^* = f(q, d_k)$, best document $d_k^* = d_k$
for each position i_j in candidate set **do**
    Insert q_{center} at position i_j in d_k to form d'_k
    Compute score $s' = f(q, d'_k)$
    if $s' > s^*$ **then**
        Update $s^* \leftarrow s'$, $d_k^* \leftarrow d'_k$
    end if
end for
return d_k^*

In this chapter, we presented three adversarial strategies for perturbing documents using the query center: `one_word_start`, `one_word_grad`/`one_word_best_grad`, and `one_word_sim`. Each method offers a distinct way to minimally and effectively influence a neural ranker's output by introducing query-aware modifications. In the

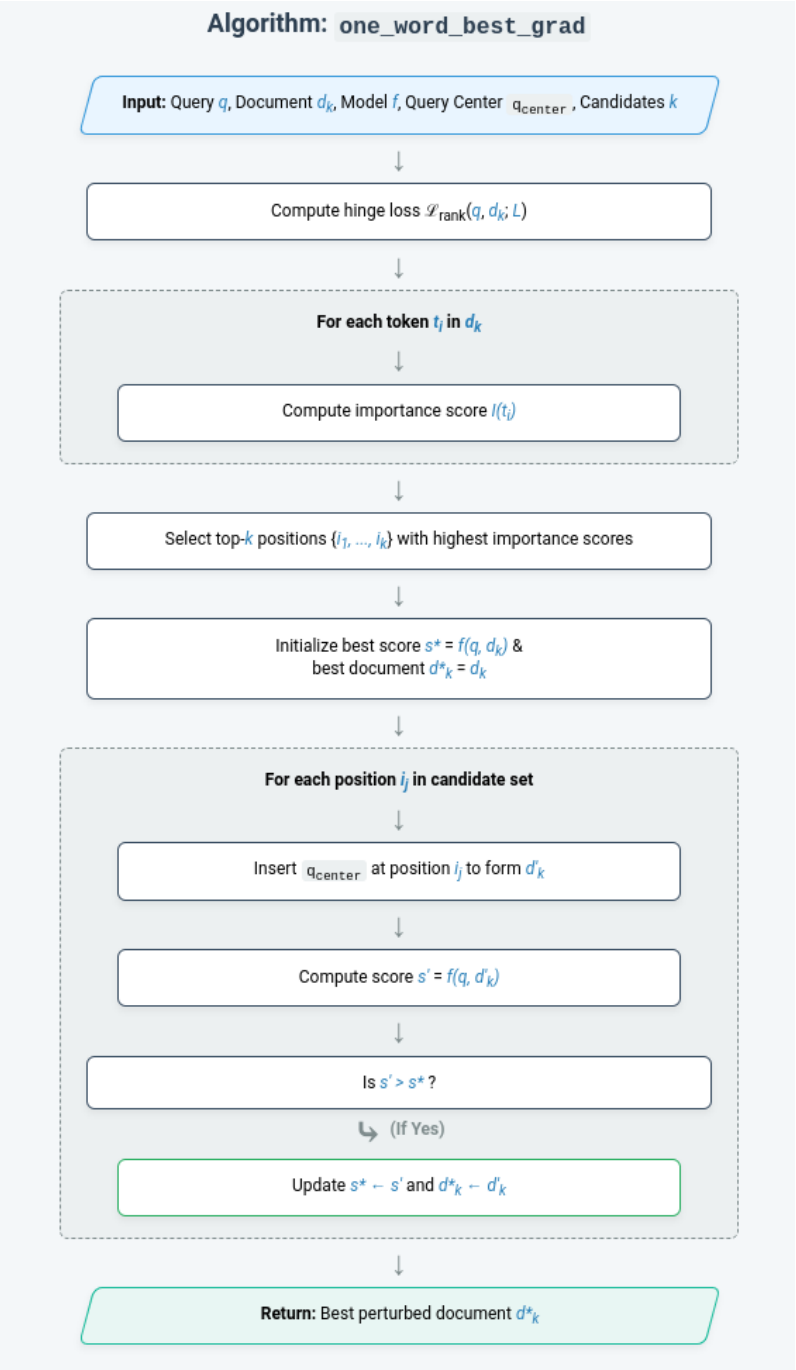


FIGURE 3.1: one_word_best_grad framework

next chapter, we evaluate these methods empirically through a series of experiments to analyze their impact on document ranking performance.

Chapter 4

Experiments

This chapter details the experimental methodology and findings of our research. We begin by describing the **dataset** utilized for our experiments in Section 4.1. Subsequently, Section 4.2 outlines the **models** implemented and evaluated. The empirical **results** of our experiments are presented and analyzed in Section 4.5. Finally, Section 4.6 discusses the key **inferences** drawn from these results and their implications.

4.1 Dataset

The primary corpus utilized for all experiments in this study is the **MS MARCO V1 passage ranking dataset**, specifically the 'msmarco-passage-full' collection. This is a large-scale dataset comprising over 8.8 million passages extracted from real web documents. It was developed by Microsoft to support research in machine reading comprehension and passage ranking, providing a substantial and diverse set of documents representative of web content. The passages are typically short and

serve as candidate answers for information needs expressed as questions or keyword queries.

For evaluating our methods, we employ query sets from three distinct and commonly used benchmarks:

- **TREC Deep Learning (DL) Track 2019 queries**: This set contains 43 queries from the TREC 2019 Deep Learning Track, which are known for their relevance and difficulty, designed to assess the performance of deep learning based retrieval models. These queries are typically informational in nature and have graded relevance judgments.
- **TREC Deep Learning (DL) Track 2020 queries**: Similar to the 2019 set, there are 54 queries from the subsequent TREC 2020 Deep Learning Track, continuing the effort to provide challenging and realistic search tasks for modern retrieval systems.
- **MS MARCO development (dev) set queries**: This set of consists of 200 sampled queries from the MS MARCO dev set.

The use of these varied query sets allows for a comprehensive evaluation of our proposed methods across different distributions and difficulty levels of information retrieval tasks.

4.2 Models

In this study, we employ and evaluate two neural ranking models from the Hugging Face model repository. Each model represents a different architectural approach to

document re-ranking in information retrieval. Their details are summarized below. Evaluation metrics are reported in Table 4.1.

- **BERT-base-mdoc-BM25 [10]**
  - **Model:** `Luyu/bert-base-mdoc-bm25`
  - **Architecture:** BERT-base (uncased)
  - **Training:** Fine-tuned on the MS MARCO document corpus with BM25 negatives, enabling it to learn re-ranking over candidates initially retrieved using a lexical retriever.
  - **Purpose:** Used to rerank passages retrieved by BM25 using semantic matching, providing a transformer-based baseline for evaluating the impact of adversarial perturbations.
- **monoT5-base-MSMARCO [11]**
  - **Model:** `castorini/monot5-base-msmarco`
  - **Architecture:** T5-base
  - **Training:** Fine-tuned on the MS MARCO passage ranking dataset in a pointwise manner where each query-passage pair is scored independently.
  - **Purpose:** Serves as a strong baseline for mono-style re-ranking, useful to assess how perturbations affect scores produced by a generative transformer architecture.

These models were chosen to represent effective and commonly used neural architectures for ranking, allowing us to assess their performance on the specified datasets and query sets. The following sections will detail the experimental setup and the results obtained using these models.

Model	trec-dl-2019 ndcg@10	MAP	trec-dl-2020 ndcg@10	MAP
BERT-base-mdoc-BM25	0.6653	0.3563	0.6378	0.3561
monoT5-base-MSMARCO	0.6995	0.3676	0.6771	0.3944

TABLE 4.1: Top 100 documents are retrieved using BM25 and reranked using the NRMs.

4.3 Implementation Details

This section outlines the technical specifications and procedures used to conduct the experiments. All implementations were carried out using Python with the PyTorch deep learning framework and the Hugging Face `Transformers` library for model access.

4.3.1 Baseline Method: PRADA

To establish a comparative baseline, we implemented the **PRADA (PRactical black-box ADversarial Attacks)** method as described by Wu et al. [1]. Since PRADA is a black-box attack, it does not require access to model gradients rather, it trains a Surrogate Model. However, due to limited computational power, we choose to use a white box model attack approach.

- **Perturbation:** The attack promotes a target document by identifying important words using the surrogate model's gradients and substituting them with synonyms.

The performance of PRADA serves as a benchmark to evaluate the effectiveness of our proposed minimal, one-word attack strategies.

4.3.2 Proposed Attack Strategies

Our proposed methods were implemented based on the algorithms presented in Chapter 3. The core component for all strategies is the **query center**, which was computed using counter-fitted word embeddings [9] to find the semantic centroid of a query.

- For the `one_word_sim` strategy, cosine similarity was used to find the closest (in terms of similarity) word in the document to the query center for replacement.
- For the gradient-based strategies, `one_word_grad` and `one_word_best_grad`, the hinge loss and importance scores were computed as per Equations 3.1 and 3.2.
- In the `one_word_best_grad` method, the number of candidate positions was set to $k = 20$. The model's scoring function was used to evaluate each of these candidate positions to find the optimal insertion point.

4.3.2.1 Evaluation Metrics

To evaluate the effectiveness and subtlety of our adversarial perturbation methods, we use the following evaluation metrics:

1. Success Rate (SR): This measures the average proportion of successful attacks across all queries. An attack is considered successful if the perturbed document is ranked higher than the original. Let S_q be the number of successful attacks and T_q be the total attacked documents for a query q. Then:

$$\text{Success Rate} = \frac{1}{|Q|} \sum_{q \in Q} \frac{S_q}{T_q} \times 100 \tag{4.1}$$

2. Interval Success Rate (ISR): The ranked list is divided into intervals, such as [11–20], [21–30], etc. For each query q, we attack all documents whose rank lies within the interval $[a, b]$. Let $S_q^{[a,b]}$ be the number of successful attacks, and $N_q^{[a,b]} = 10$ the number of attacked documents in the interval. The average interval success rate is:

$$\text{IntervalSR}^{[a,b]} = \frac{1}{|Q|} \sum_{q \in Q} \frac{S_q^{[a,b]}}{N_q^{[a,b]}} \tag{4.2}$$

3. Average Similarity Score(SS): For each document and perturbed document pair (d_k, d'_k), let $\text{sim}(d_k, d'_k) \in [0, 1]$ denote the similarity score. Then:

$$\text{Average Similarity} = \frac{1}{N} \sum_{i=1}^{N} \text{sim}(d_i, d'_i) \tag{4.3}$$

4. Perturbation Percentage (PP): This measures the percentage of modified tokens relative to the document length:

$$\text{Perturbation \%} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{p_i}{|d_i|} \times 100 \right) \tag{4.4}$$

where p_i is the number of perturbed tokens and $|d_i|$ is the length of document d_i.

5. Perturbation Number (PN): This is the average number of words perturbed per document:

$$\text{Perturbation Number} = \frac{1}{N} \sum_{i=1}^{N} p_i \tag{4.5}$$

6. Rank boost (RB): For any document d_k and perturbed document d'_k, from a ranked list L, the rank boost is defined by

$$Rank_L(d'_k) - Rank_L(d_k) \tag{4.6}$$

Then we take the average of the rank boosts of all the documents.

7. Score boost (SB): For any document d_k and perturbed document d'_k, from a ranked list L, the score boost is defined by

$$f(q, d'_k) - f(q, d_k) \tag{4.7}$$

where q is the query, f is the Neural Ranker Model. Then we take the average of the score boosts of all the documents.

These metrics collectively evaluate the trade-off between effectiveness (in rank promotion) and subtlety for each perturbation method.

4.4 Evaluation Setup

To assess the effectiveness of our proposed adversarial strategies, we conducted experiments on both the `BERT-base-mdoc-BM25` and `monoT5-base-MSMARCO` models. The evaluation protocol was designed as follows:

1. **Query and Document Selection:** We used 43 queries from trec-dl-2019, 54 queries from trec-dl-2020, and a set of 200 queries sampled from the MS MARCO development set. For each query, the top 100 passages were retrieved using BM25 and subsequently re-ranked by the target NRM.

2. **Target for Attack:** For each ranked list of 100 documents, all the documents from rank 11 to 100 are selected for attack. We left out the first 10 documents as they are highly relevant to the query and unlikely to get promoted even further, as suggested in [1].

3. **Metrics:** The performance of each attack strategy was evaluated using the metrics defined in Section 4.3.2.1.

The results comparing our proposed minimal one-word attack strategies against the PRADA baseline are presented separately for each target model.

4.5 Results

The key results from our experiments with various models for each of the proposed methods 3 are presented in this section. For each set of queries (trec-dl-2019, trec-dl-2020, and msmarco-dev-200) and for each of the described 4.2 models we have produced the results using the above metrics 4.3.2.1. As PRADA [1] trains a surrogate model, but we are doing a white box attack (in case of one_word_grad and one_word_best_grad) for a fair comparison, we have reproduced the results of PRADA for the same models 4.2, considering them as white-box.

For the interval success rate consider the interval $I(k) = [10k + 1, 10k + 10]$ for $k = 1, 2, \cdots, 9$ (See tables 4.3, 4.5, 4.7, 4.9, 4.11).

Model 1: BERT-base-mdoc-BM25

The results for the model BERT-base-mdoc-BM25 with various sets of queries from the MS MARCO V1 passage ranking dataset are given.

TABLE 4.2: Attack performance on the `BERT-base-mdoc-BM25` model on **trec-dl-2019** queries

Attack Method	SR (%)	SS	PP (%)	PN	RB	SB
PRADA (Baseline)	96.6931	0.8743	12.3033	8.2534	17.8460	1.9885
`one_word_start`	56.6402	0.9685	0.9888	0.6375	8.2058	1.0864
`one_word_sim`	38.0952	**0.9822**	**0.8112**	**0.5219**	4.5992	0.5642
`one_word_grad`	68.2275	0.9710	1.4688	0.9481	5.4693	0.6073
`one_word_best_grad`	90.9788*	0.97428*	1.4948*	0.9671	12.9751*	1.6171*

TABLE 4.3: Interval Success Rate (%) on the `BERT-base-mdoc-BM25` model on **trec-dl-2019** queries.

Attack Method	I(1)	I(2)	I(3)	I(4)	I(5)	I(6)	I(7)	I(8)	I(9)
PRADA (Baseline)	89.5238	95.0000	96.9047	97.8571	98.8095	97.6190	98.8095	99.0476	96.6666
`one_word_start`	50.0000	54.5238	57.1428	57.8571	57.8571	62.3809	60.2380	56.9047	52.8571
`one_word_sim`	26.9047	35.0000	39.0476	38.3333	44.5238	39.7619	45.4761	39.2857	34.5238
`one_word_grad`	53.0952	65.4761	72.8571	73.5714	74.5238	74.7619	75.9523	70.9523	52.8571
`one_word_best_grad`	82.1428	89.2857	94.7619	94.2857	94.2857	92.3809	93.3333	94.0476	84.2857

TABLE 4.4: Attack performance on the `BERT-base-mdoc-BM25` model on **trec-dl-2020** queries

Attack Method	SR (%)	SS	PP (%)	PN	RB	SB
PRADA (Baseline)	97.7377	0.8524	13.8311	9.2889	20.0963	2.4173
`one_word_start`	44.1791	0.9781	0.8137	0.5270	4.2037	0.5251
`one_word_sim`	30.8988	**0.9819**	**0.7037**	**0.4435**	2.0583	0.2421
`one_word_grad`	65.0899	0.9657	1.4246	0.9204	3.6481	0.3979
`one_word_best_grad`	87.1020*	0.9685*	1.4507*	0.9379*	9.4728	1.1624

TABLE 4.5: Interval Success Rate (%) on the `BERT-base-mdoc-BM25` model on **trec-dl-2020** queries.

Attack Method	I(1)	I(2)	I(3)	I(4)	I(5)	I(6)	I(7)	I(8)	I(9)
PRADA (Baseline)	92.9629	95.9183	98.4905	99.2452	99.0566	98.8679	99.4339	99.6226	98.6792
`one_word_start`	35.5555	42.3005	40.9433	44.1509	45.2830	48.8679	46.4150	50.0000	42.4528
`one_word_sim`	21.6666	29.3135	33.2075	32.8301	32.6415	32.0754	36.7924	35.4716	24.9056
`one_word_grad`	52.8301	63.9622	66.6037	66.6037	72.4528	71.8867	68.1132	68.5606	54.2635
`one_word_best_grad`	78.8461	85.0000	86.9230	87.5000	90.5769	90.7692	91.1538	90.3846	82.7450

TABLE 4.6: Attack performance on the `BERT-base-mdoc-BM25` model on **msmarco-dev-200** queries

Attack Method	SR (%)	SS	PP (%)	PN	RB	SB
PRADA (Baseline)	95.9279	0.8470	13.9678	9.5703	22.1350	2.5615
`one_word_start`	41.5759	0.9801	0.7562	0.4905	4.7400	0.5544
`one_word_sim`	29.3083	**0.9830**	**0.6486**	**0.4226**	2.5354	0.2707
`one_word_best_grad`	81.3605*	0.9720*	1.3621*	0.8916*	9.5666	1.1307

TABLE 4.7: Interval Success Rate (%) on the `BERT-base-mdoc-BM25` model on **msmarco-dev-200** queries.

Attack Method	I(1)	I(2)	I(3)	I(4)	I(5)	I(6)	I(7)	I(8)	I(9)
PRADA (Baseline)	90.9954	93.8407	95.8648	95.7575	96.9696	97.2713	97.4111	97.9187	98.5204
`one_word_start`	32.2448	37.0408	41.2755	42.0408	43.7755	44.9489	46.0204	44.7959	42.0408
`one_word_sim`	21.8367	25.6122	28.7755	31.4795	31.5306	32.5000	32.8571	32.2448	26.9387
`one_word_best_grad`	71.2755	78.4183	81.8367	83.9285	83.9285	85.2040	86.3775	85.7653	75.5102

Model 2: monoT5-base-MSMARCO

The results for the model monoT5-base-MSMARCO with various sets of queries from the MS MARCO V1 passage ranking dataset is given.

TABLE 4.8: Attack performance on the `monoT5-base-MSMARCO` model on **trec-dl-2019** queries

Attack Method	SR (%)	SS	PP (%)	PN	RB	SB
PRADA (Baseline)	69.7883	0.8948	9.70312	6.6714	11.6185	1.3992
`one_word_start`	57.4867	0.9773	0.9632	0.6087	9.2179	1.2715
`one_word_sim`	53.7830	**0.9811**	**0.9116**	**0.5722**	7.4515	0.9832
`one_word_best_grad`	65.1851*	0.9744*	1.1536*	0.7346*	**11.7809**	**1.5384**

TABLE 4.9: Interval Success Rate (%) on the `monoT5-base-MSMARCO` model on **trec-dl-2019** queries

Attack Method	I(1)	I(2)	I(3)	I(4)	I(5)	I(6)	I(7)	I(8)	I(9)
PRADA (Baseline)	66.1904	69.7619	69.7619	71.1904	75.0000	72.8571	67.8571	69.5238	65.9523
`one_word_start`	55.0000	60.2380	63.3333	59.5238	61.4285	57.1428	55.7142	56.6666	48.3333
`one_word_sim`	52.6190	56.9047	60.7142	55.9523	58.0952	54.0476	49.7619	51.4285	44.5238
`one_word_best_grad`	59.5238	68.0952	66.1904	67.6190	67.1428	69.5238	66.1904	66.6666	55.7142

TABLE 4.10: Attack performance on the `monoT5-base-MSMARCO` model on **trec-dl-2020** queries

Attack Method	SR (%)	SS	PP (%)	PN	RB	SB
PRADA (Baseline)	59.6226	5.2442	3.5530	50	7.4280	0.9449
`one_word_start`	50.2306	0.9772	0.8779	0.5522	5.5786	0.7081
`one_word_sim`	45.5345	**0.9815**	**0.8085**	**0.5081**	4.3475	0.5244
`one_word_best_grad`	59.4968*	0.9693*	1.1018*	0.7054*	**7.9507**	**1.0019**

TABLE 4.11: Interval Success Rate (%) on the `monoT5-base-MSMARCO` model on **trec-dl-2020** queries

Attack Method	I(1)	I(2)	I(3)	I(4)	I(5)	I(6)	I(7)	I(8)	I(9)
PRADA (Baseline)	54.9056	61.8867	60.3773	61.1320	60.9433	62.6415	58.3018	62.2641	54.1509
`one_word_start`	48.1132	53.5849	55.8490	55.2830	48.8679	53.5849	50.1886	46.9811	39.6226
`one_word_sim`	45.2830	51.1320	52.8301	50.3773	46.2264	47.9245	42.8301	38.8679	34.3396
`one_word_best_grad`	53.9622	62.2641	62.8301	62.0754	59.8113	64.5283	61.3207	57.9245	50.7547

4.6 Inferences

The experimental results presented in Section 4.5 provide several critical insights into the behavior of Neural Ranking Models (NRMs) under adversarial attacks. The following inferences can be drawn from our analysis:

1. **High Efficacy of Minimalist, Single-Word Attacks:** A primary finding is that NRMs exhibit significant vulnerability to extremely minimal perturbations. Our `one_word_best_grad` method, which introduces only a single word into the document, achieves a success rate (SR) that is highly competitive with the PRADA [1] baseline. For instance, on the trec-dl-2019 query set (Table 4.2), `one_word_best_grad` recorded an SR of 90.98% by perturbing just one word (PN $\approx$ 0.97), whereas the baseline PRADA required modifying over eight words on average (PN $\approx$ 8.25) to achieve its 96.69% SR. This trend is followed by the other query sets and the monoT5 model too, in fact for the model monoT5, in some cases our method `one_word_best_grad` **outperforms** PRADA in terms of RB and SB (see tables 4.8 and 4.10). This demonstrates that the semantic integrity of a document's ranking score can be compromised by a single, strategically placed, query-relevant term, highlighting a crucial efficiency in our attack and a significant vulnerability in the model.

2. **The Critical Role of Perturbation Placement:** The performance gap between our proposed methods underscores the importance of the attack strategy. The gradient-based methods (`one_word_grad` and `one_word_best_grad`) consistently and significantly outperform the heuristic-based approaches (`one_word_start` and `one_word_sim`). This suggests that merely inserting a query-relevant word is not sufficient; its placement within the document is also

important. The superior performance of `one_word_best_grad` over `one_word_grad` further refines this insight, indicating that leveraging gradient information to identify a candidate set of sensitive locations, and then selecting the optimal one, is a highly effective strategy for maximizing rank promotion.

3. **A "Goldilocks Zone" for Attack Success:** The Interval Success Rate (ISR) analysis (as seen in Tables 4.3, 4.5, 4.7, 4.9 and 4.11) reveals a distinct pattern resembling a normal distribution or bell curve (see figures 4.1 and 4.2 corresponding to the tables 4.3 and 4.11 respectively). The attack success rate is not uniform across the ranked list.

   - **Low Success in High-Ranked Intervals (e.g., 11-30):** Documents in the top ranks are already considered highly relevant by the NRM. Adding a single word provides only a marginal benefit, as their relevance scores may already be approaching a saturation point. It is difficult to achieve a significant rank boost when a document is already near the top.
   - **Peak Success in Mid-Ranked Intervals (e.g., 40-80):** The highest success rates are observed for documents in the middle of the ranked list. These documents possess a moderate level of relevance, making them highly susceptible to manipulation. A single, well-placed word can provide the necessary semantic signal to dramatically increase their relevance score and propel them up the ranks.
   - **Low Success in Low-Ranked Intervals (e.g., 91-100):** Documents at the bottom of the list are often fundamentally irrelevant to the query. The semantic gap is too wide for a single-word perturbation to bridge effectively. Consequently, the attack is less likely to succeed in promoting these documents to a substantially higher rank.

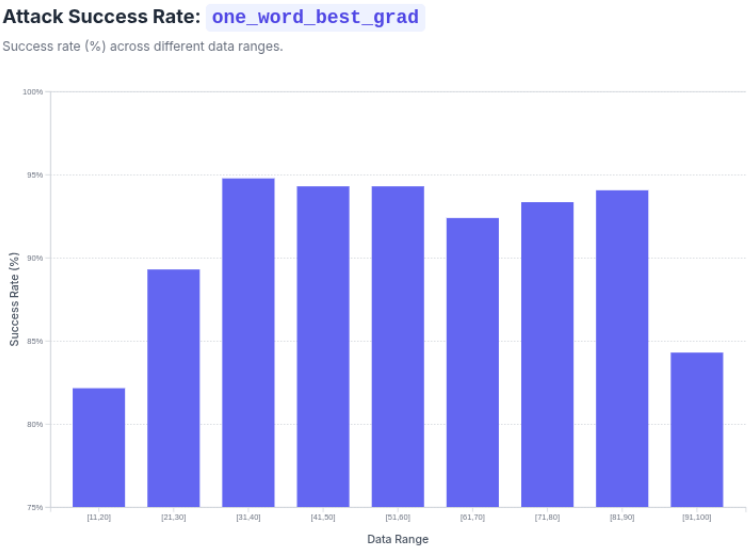


FIGURE 4.1: Interval success rate of one_word_best_grad method on BERT-base-mdoc-BM25, trec-dl-2019 queries, data from Table 4.3.

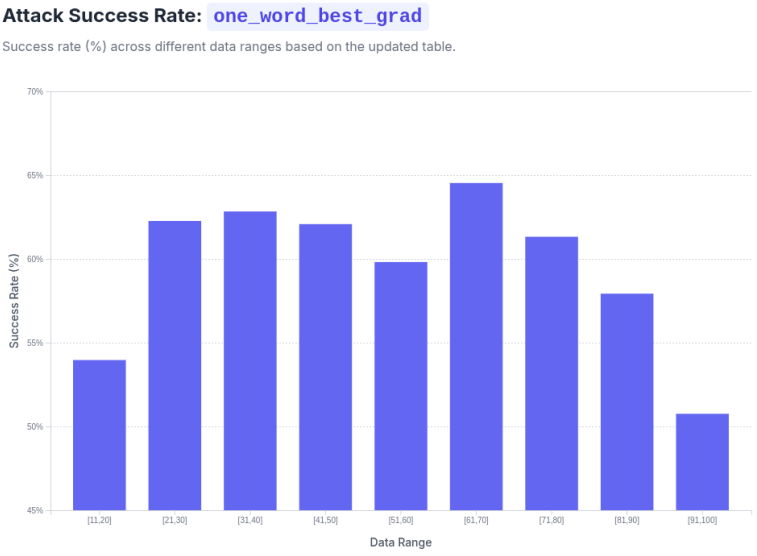


FIGURE 4.2: Interval success rate of one_word_best_grad method on monoT5-base-MSMARCO, trec-dl-2020 queries, data from Table 4.11.

In summary, our experiments demonstrate that while NRMs are powerful, they are also brittle. Their sensitivity can be exploited by minimalist, query-aware attacks, particularly when targeting documents within a specific middle range of relevance.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this dissertation we have investigated the robustness of modern Neural Ranking Models (NRMs) against minimal, word-level adversarial attacks. As these models become increasingly integral to real-world information retrieval systems, understanding their vulnerabilities is of paramount importance. Our work focused on a highly constrained attack scenario: promoting a document's rank by adding or substituting just a single, query-relevant word.

We introduced a series of novel attack strategies, centered around the concept of a "query center"—a word that semantically represents the core intent of a user's query. Our proposed methods ranged from simple heuristics, like inserting the query center at the start of a document (`one_word_start`), to more sophisticated, white-box techniques that leverage model gradients to identify optimal perturbation locations (`one_word_grad` and `one_word_best_grad`).

Our experiments, conducted on standard benchmark datasets (TREC-DL 2019, TREC-DL 2020, and MS MARCO DEV queries with MSMARCO PASSAGE V1 data set), yielded significant findings. The primary conclusion is that state-of-the-art NRMs, such as BERT-based re-rankers, are remarkably brittle. Our most effective strategy, `one_word_best_grad`, successfully promoted documents with a very high success rate (up to 90.98%), rivaling a complex baseline (PRADA) that required modifying significantly more words. This demonstrates that an attacker can achieve their objective with extremely subtle and efficient perturbations, posing a realistic threat.

Furthermore, our analysis of attack success across different rank intervals revealed a "Goldilocks Zone." Attacks were most effective on mid-ranked documents, which are moderately relevant, and less effective on highly-ranked (already relevant) or low-ranked (highly irrelevant) documents. This provides a nuanced understanding of where NRMs are most susceptible to manipulation.

In summary, this research successfully demonstrates that the complex relevance signals learned by NRMs can be effectively manipulated through minimal, targeted, and query-aware word additions. The findings highlight a critical trade-off between the expressive power of these models and their adversarial robustness.

5.2 Future Work

The insights and methodologies developed in this dissertation open up several promising avenues for future research. The following directions could build upon our work:

1. **Developing Robust Defense Mechanisms:** The most critical next step is to use the identified vulnerabilities to build more resilient NRMs. Adversarial

training, where models are retrained on a diet of perturbed examples generated by methods like `one_word_best_grad`, could be explored. This would test whether NRMs can learn to become insensitive to such minimal manipulations.

2. **Transitioning to Black-Box Scenarios:** While our gradient-based methods proved most effective, they operate in a white-box setting. A significant research challenge would be to adapt the `one_word_best_grad` strategy to a practical black-box scenario. This might involve training a surrogate model to estimate gradients or developing a query-efficient search algorithm to find the optimal insertion point without direct access to the model's internals.

3. **Exploring Multi-Word, Minimal Attacks:** Our work focused on the extreme case of a single-word attack. Future research could explore the trade-off between the number of perturbed words and attack success. For instance, an attack that intelligently adds 2-3 words might achieve a near-perfect success rate while remaining far more subtle than existing baselines.

4. **Analysis of Cross-Model and Cross-Domain Transferability:** An interesting investigation would be to determine if the adversarial tokens and positions identified for one NRM are effective against another. High transferability would suggest a more fundamental weakness in the underlying Transformer architecture or pre-training data, rather than in a specific model's fine-tuning.

5. **Extending to Other Retrieval Tasks:** The methodologies proposed here could be adapted and tested on related tasks, such as conversational search, question answering, and recommendation systems, to assess the broader impact of such vulnerabilities across the information access landscape.

6. **Applications in Explainable IR (XAI-IR):** The minimalist nature of our framework can be repurposed as a diagnostic tool for model interpretability.

The identified `query_center` reveals the specific term a model deems most representative of a query's semantics. Crucially, the optimal insertion position found by `one_word_best_grad` pinpoints the exact locations to which the model is most sensitive. By observing where a query-relevant term maximally boosts the score, we can generate post-hoc explanations for why a model considers a document relevant, thereby making the NRM's decision-making process more transparent.

By pursuing these directions, the research community can work towards developing next-generation information retrieval systems that are not only highly effective but also secure and trustworthy.

References

[1] C. Wu, R. Zhang, J. Guo, M. de Rijke, Y. Fan, X. Cheng, Prada: Practical black-box adversarial attacks against neural ranking models, arXiv:2204.01321 (2023).

[2] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks., arXiv:2004.01970. (2013).

[3] J. Su, D. V. Vargas, K. Sakurai, One pixel attack for fooling deep neural networks, IEEE Transactions on Evolutionary Computation 23 (2019) 828–841. URL: http://dx.doi.org/10.1109/TEVC.2019.2890858. doi:10.1109/tevc.2019.2890858.

[4] J. Gao, J. Lanchantin, M. L. Soffa, Y. Qi, Black-box generation of adversarial text sequences to evade deep learning classifiers, arXiv:1801.04354 (2018).

[5] J. Ebrahimi, A. Rao, D. Lowd, D. Dou, HotFlip: White-box adversarial examples for text classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 31–36. doi:10.18653/v1/P18-2006.

[6] D. Pruthi, B. Dhingra, Z. C. Lipton, Combating adversarial misspellings with robust word recognition, 2019.

[7] Y. Wang, L. Lyu, A. Anand, Bert markers are brittle: A study using adversarial perturbations, arXiv:1204.11714 (2021).

[8] M. Ivgi, J. Berant, Achieving model robustness through discrete adversarial training neural ranking models, arXiv:2104.05062 (2021).

[9] N. Mrksi˘c´, D. O. S. ´, B. Thomson, M. Gasi˘c´, L. Rojas-Barahona, P.-H. Su, D. Vandyke, T.-H. Wen, S. Young, Counter-fitting word vectors to linguistic constraints, arXiv:1603.00892 (2016).

[10] L. Gao, Z. Dai, J. Callan, Rethink training of bert rerankers in multi-stage retrieval pipeline, in: The 43rd European Conference On Information Retrieval (ECIR), 2021.

[11] R. Nogueira, Z. Jiang, R. Pradeep, J. Lin, Document ranking with a pretrained sequence-to-sequence model, in: T. Cohn, Y. He, Y. Liu (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online, 2020, pp. 708–718. URL: https://aclanthology.org/2020.findings-emnlp.63/. doi:10.18653/v1/2020.findings-emnlp.63.