Black-Box Transformation of the GASLITE Attack

Ishay Yemini

Tel Aviv University

September 25, 2025

ABSTRACT

Retrieval-Augmented Generation (RAG) systems are increasingly vulnerable to adversarial attacks that manipulate their retrieval components. However, many state-of-the-art attacks require white-box access to model gradients, an assumption that is often unrealistic in practice. This project introduces the Combination Attack, a highly effective black-box attack on dense retrieval models. My method employs a multi-stage optimization strategy: it first uses a random search to generate a promising initial adversarial passage, which is then refined using a robust, block-based optimization algorithm inspired by the Square Attack. I evaluate my attack on the MSMARCO [2] dataset against a diverse suite of nine modern sentence-transformer models. The results demonstrate that the Combination Attack significantly outperforms existing black-box baselines. Most notably, it achieves top-1 retrieval success rates of over 90% for most models, demonstrating performance remarkably close to that of the state-of-the-art white-box attack, GASLITE. This work shows that the performance gap between black-box and white-box attacks is smaller than previously assumed, highlighting a practical and potent security vulnerability in modern RAG systems that rely on model obscurity for defense. [1]

1 INTRODUCTION

Retrieval-augmented generation (RAG) systems are becoming more and more common in various natural language processing applications. This has led to a rise in research dedicated to understanding and mitigating their vulnerabilities to adversarial attacks. These attacks aim to degrade the performance of RAG systems, often by manipulating the retrieval component. However, many existing attack methods operate under the assumption of white-box access, implying that the attacker has complete knowledge of the RAG system's internal mechanisms, including its parameters and gradients. This assumption is often unrealistic in real-world scenarios, where attackers typically lack such privileged access.

GASLITE is one such attack that, based on its nature of manipulating embeddings through gradient information, requires direct access to the gradients of the dense retrieval model. To address the limitations imposed by white-box requirements, this project seeks potential methods and research directions for transforming the GASLITE attack into a black-box attack. The focus will be on techniques that can bypass the need for direct gradient access, thereby enhancing the practicality and applicability of adversarial attacks against dense retrieval systems in more realistic threat models.

2 RELATED WORK

2.1 GASLITE

GASLITE is a newly proposed attack method against embedding-based RAGs [3]. GASLITE uses a gradient-based search method in order to generate adversarial passages for promoting a chosen malicious passage, without modifying the model or relying on the specific corpus content. By carefully crafting perturbations to add on top of the selected poison, GASLITE can promote it to the top of the retrieval list for specific queries [3].

In this paper, the researches focused on three types of attacks: Knows All, where the attacker knows the specific query to be attacked, in which case they consistently managed to make the crafted passages be the top-1 result; Knows What, targeting a concept and not a specific query, where they also achieved impressive results, managing the promote the poison to the top-10 for most queries and most models; and Knows Nothing, for concept agnostic attacks, which they found to be the most challenging but still achieved top-10 visibility for more than 10% of queries [3].

Still, the fundamental limitation of GASLITE, which I seek to improve on in my research, is its requirement for direct access to the model's gradients. This limits the attack to White-Box scenarios only, a limit that is often not met in practical settings where RAG systems are deployed as black boxes with none (or limited) access to their gradients and internal workings [3].

2.2 CorruptRAG

Zhang et al. propose CorruptRAG, a poisoning attack designed for practicality and stealth [17]. The key innovation of CorruptRAG is its ability to succeed by injecting only a single poisoned document into the knowledge base, whereas other methods often required multiple injections to ensure retrieval.

The attack's primary goal is to corrupt the final generated answer of the RAG system, which it achieves through two distinct strategies: "Answer Suppression", which aims to prevent the LLM from finding the correct answer in the retrieved context, and "Answer Knowledge Corruption", which subtly misleads the LLM into trusting the incorrect information provided in the poisoned document [17].

While my Combination Attack also operates with a single document, its objective and evaluation are focused on maximizing retrieval rank. In contrast, CorruptRAG's techniques are centered on linguistic manipulation to influence the LLM's reasoning, with its success being measured by the incorrectness of the final generated output. Additionally, this attack could not be easily adapted into boarder settings such as "Knows-What" [3], where more queries need to be optimized simultaneously.

---

[1]My attack code is available at https://github.com/ishayyemini/combination_attack

2.3 PoisonedRAG

Zou et al. introduce PoisonedRAG, a comprehensive knowledge corruption attack that targets the full RAG pipeline [20]. Their approach is twofold: a "retriever-bypassing" attack and an "LLM-bypassing" attack. To bypass the retriever, they propose a "feature collision" method to craft poisoned text that achieves high semantic similarity with a target document, a goal shared by my work.

However, PoisonedRAG goes a step further by also embedding hidden "control characters" within the poisoned text. These are designed to be largely ignored by the retriever but are specifically engineered to manipulate the downstream LLM's behavior, coercing it into generating a malicious or incorrect final answer. This dual-focus on both retriever and LLM manipulation distinguishes it from my work, which concentrates exclusively on compromising the integrity of the retrieval stage [20].

2.4 Square Attack

Andriushchenko et al. [1] introduce the Square Attack, a query-efficient black-box adversarial attack. The method utilizes a randomized search approach that applies localized, square-shaped perturbations to an image. The attack starts with a pre-defined square size, chooses a random area to attack, and then on each iteration the square's area gets smaller. This strategy allows the attack to be much faster and efficient than random search, as it counts not just a single pixel but an entire area. The authors managed to demonstrate that the Square Attack can even outperform some white-box attacks on standard benchmarks, and is capable of breaking defenses that appear robust to traditional PGD attacks.

2.5 Universal Adversarial Triggers

My main direction to explore was the idea of Universal Adversarial Triggers (UATs) [13]. This idea relates to the concept of manipulating text embeddings through certain token sequences. Specifically, UATs are a sequence of tokens that, when appended to any input text, can cause a ML model to produce an incorrect or targeted prediction. The work in this area introduces gradient-guided methods for discovering these triggers. While the process for finding UATs often relies on having white-box access to the model, the resulting UATs are input-agnostic and have been shown to sometimes transfer affectively to other models, even with different embeddings and architectures [13].

UATs have been shown to demonstrate potential in attacking the safeguards of LLMs [8]. UATs might achieve this effect by approximating semantic information within their adversarial training region [12]. This implies that the manipulation cause by UATs may not be arbitrary, but could be related to semantically meaningful directions in the embedding space, similar to the embedding manipulation goals of the GASLITE attack [3]. Therefore, I wanted to explore the possibility of finding UATs that, when appended to queries or documents, can manipulate the retrieval rankings in dense retrieval models without direct access to the gradients. This transferability property of UATs, where a trigger found from a surrogate model and might also work on the target dense retriever, is especially relevant for my black-box goal.

3 TECHNICAL APPROACH

The approach of my attack evolved significantly through an ever-changing process of experimentation, analysis, and refinement, starting with a simple, random algorithm and moving towards a more sophisticated, multi-stage attack strategy.

My goal was to develop a black-box attack, capable of manipulating the retrieval ranking for a given query, with the same threat model as the "Knows-All" setting from the GASLITE paper [3]. This setting assumes the attacker knows the target query, and can craft a specific suffix to a random malicious passage info, to turn it into an adversarial passage.

3.1 Random Search Attack

The first iteration of the attack was a simple, greedy, token-by-token random search, implemented in the random_attack method. The logic was simple:

(1) Start with a base "poison".
(2) At each step, generate a pool of several hundred random candidate tokens from the model's vocabulary.
(3) For each candidate token, calculate the cosine similarity when temporarily appending it to the current passage.
(4) Select the token that yields the highest similarity score, permanently append it to the passage, and proceed to the next step.

This simple attack showed some promising success with certain passages from MSMARCO [2] using the E5 model [14], however when generalized to a 50-query trial in the same setting as GASLITE [3], the attack's success became minimal, managing to bring the poison to the top result of the query only once.

3.2 Query Stuffing

As a temporary improvement, I made an enhancement to the random attack by first appending the text of the target query directly to the end of the poison passage, before running the attack. My hypothesis was that this would immediately ground the poisoned passage to a more correct semantic region, giving the attack algorithm a stronger starting point. This simple change proved effective, allowing for almost a 62% success rate on the same top-1, 50-query trial.

3.3 Square Attack Adaptation

While query stuffing improved results, this had a major drawback, as this relied too heavily on one specific query, and would render this attack impossible to implement for the Knows-What and Knows-Nothing settings [3]. As such, a better, more powerful algorithm was needed.

Inspiration was drawn from the "Square Attack" [1], however I needed to implement this attack to our 1D setting. To do this, instead of looking at squares of pixel, I focused on segments of tokens, while trying to preserve as much of the original logic as possible. This adapted square attack works as follows:

(1) **Initialization**: The attack begins by appending a sequence of randomly chosen tokens to the base info passage.
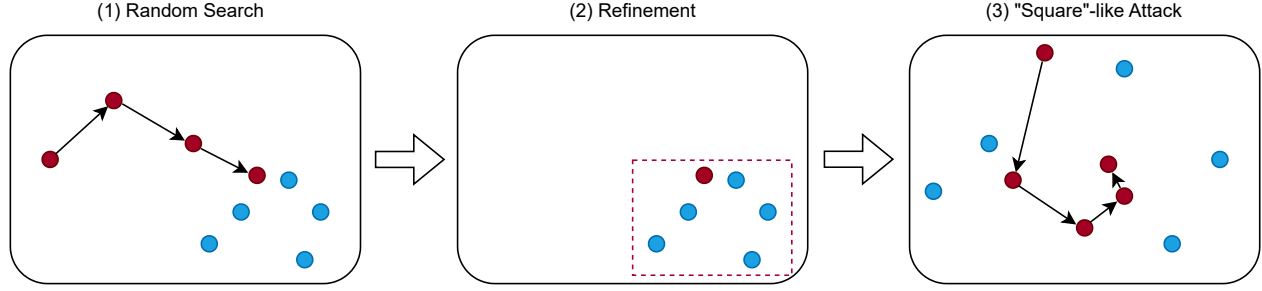(2) **Iterative Refinement**: For a set number of iterations:

Figure 1: Representation of the Combination Attack

(a) **Segment Selection**: The algorithm randomly chooses a contiguous segment of tokens within the suffix, the size of which decreases as the algorithm matures, allowing for vast exploration at the beginning and fine-tuning later on.

(b) **Block Replacement**: Multiple candidate replacements are generated for the chosen block, each of them is evaluated by calculating its similarity to the target query. If the best candidate passage has a higher similarity score than the current, it is accepted. Otherwise, the change is discarded.

(3) **Early Stopping**: If no improvement is found for a specified number of iterations, the attack terminates.

This method proved more effective than the simple random search, as it modifies multiple tokens at once, enabling it to escape local optima by making larger "jumps" in the semantic space instead of smaller, one-worded "jumps".

3.4 Combination Attack

My breakthrough came from combining both attacks, the random search attack with the square attack. My newly proposed Combination Attack is a multi-stage process, as can be seen above in Figure 1:

(1) **Random Search**: First, the simple random attack is run to generate an initial rough sequence of tokens.

(2) **Refinement**: Then, the token sequence generated is fed to the next stage to refocus the attack area.

(3) **"Sqaure"-like Attack**: Finally, the tokens are used as the starting point for a square attack.

This hybrid approach leverages the strengths of both methods. The random search allows for bolder movements in the semantic space, essentially placing the initial passage in a promising semantic region. Then, the square attack takes over, performing a more comprehensive local search to refine this starting point into a highly optimized final passage. This allows it to "snake" around the specific semantic region, precisely refining it to align with the target embedding.

3.5 Hyperparameter Optimization

With the powerful attack algorithm in place, my final step was to tune its hyperparameters. The script `comb_hparam_search.py`

was used to perform a random search over the key parameters of Combination Attack:

- `total_tokens`: The number of tokens to append.
- `p_init`: The initial "window" size.
- `num_iters`: The number of optimization iterations.
- `random_pool_per_pos`: The diversity of random tokens to be sampled at each position.

This search was run twice, each time for dozens of trials, when the second time refined the search over a narrower space. The objective was to maximize the average cosine similarity, over a chosen model, E5 [14].

Additionally, a similar search was run to find the best hyperparameters for the `adversarial decoding` attack.

All hyperparameters for both attacks are detailed in Section 5.1 below.

4 THREAT MODEL

Exactly as with the Knows-All setting in GASLITE [3], I attacked a single query with one adversarial passage, meaning I tried to get a suffix-controlled text as similar as possible to a random query. I averaged results on 50 randomly sampled queries, from both MS-MARCO [2] and NQ [7].

The attack was built as a black-box attack, which meant it had access to the tokenizer and the embedding, but otherwise it had no access to any other part of the model, such as the gradients.

5 RESULTS

5.1 Experimental Setup

I attacked an extensive setup to test popular retrievers, and compared my results to stuffing and GASLITE baselines.

Models. I evaluated many embedding-based retreivers, mainly to compare them to existing benchmarks in GASLITE [3]. The models I attacked are E5 [14], MiniLM [15], GTR-T5 [10], aMPNet [11], Arctic [9], Contriever [5], Contriever-MS [5], ANCE [16], mMPNet [11].

Datasets. I ran the experiment on two datasets, MSMARCO [2] and NQ [7]. MSMARCO contains a corpus of 8.8M passages and 0.5M search queries, and is a standard dataset used to evaluate RAG attacks. NQ is another database I used, mainly to compare to

Table 1: Experiment Results

| Dataset | Sim. | Model | appeared@10 (appeared@1) | | | | objective | | | |
			info Only	stuffing	Adv. Dec.	Combi	info Only	stuffing	Adv. Dec.	Combi
MSMARCO	cosine	E5	0% (0%)	18% (2%)	56% (14%)	**100% (96%)**	0.674	0.839	0.884	**0.948**
		MiniLM	0% (0%)	18% (6%)	66% (22%)	**100% (94%)**	0.017	0.584	0.734	**0.923**
		GTR-T5	0% (0%)	60% (24%)	84% (50%)	**100% (100%)**	0.386	0.772	0.828	**0.910**
		aMPNet	0% (0%)	38% (6%)	68% (28%)	**100% (92%)**	0.005	0.589	0.735	**0.905**
		Arctic	0% (0%)	86% (56%)	82% (20%)	**100% (100%)**	0.167	0.605	0.547	**0.763**
	dot	Contriever	0% (0%)	98% (57%)	88% (54%)	**100% (100%)**	0.485	1.418	1.382	**2.463**
		Contriever-MS	0% (0%)	56% (22%)	80% (48%)	**100% (100%)**	0.475	1.607	1.773	**2.799**
		ANCE	0% (0%)	24% (4%)	90% (42%)	**100% (100%)**	698.202	710.119	714.435	**717.496**
		mMPNet	0% (0%)	38% (8%)	90% (42%)	**100% (100%)**	6.527	27.076	33.468	**40.336**
NQ	cosine	E5	0% (0%)	44% (14%)	96% (30%)	**100% (100%)**	0.642	0.837	0.869	**0.941**
		MiniLM	0% (0%)	40% (16%)	88% (54%)	**100% (98%)**	-0.011	0.593	0.741	**0.936**
	dot	Contriever-MS	0% (0%)	62% (36%)	98% (70%)	**100% (100%)**	0.480	1.621	1.805	**2.734**
		ANCE	0% (0%)	42% (18%)	98% (74%)	**100% (100%)**	698.727	712.009	715.471	**718.557**

GASLITE [3] and prove that the attack generalizes to other datasets, which contains a corpus of 2.7M passages and 3.5K search queries. For the malicious info, I sampled a random passage from ToxiGen [4].

My Attack. The hyperparameters used in my attack were calibrated to achieve the highest success, and were `total_tokens=72`, `p_init=0.3946`, `num_iters=2940`, and `random_pool_per_pos=316`.

Other Attacks. For a control, I tested **info Only** where the adversarial passage is just the chosen info alone. Then, for comparison, I used **stuffing**, where the query was appended to the end of info, and **Adv. Dec.** [18], the implementation of which I recreated in order to fit within our threat model, and used the following hyperparameters: `max_tokens=57`, `beam_size=9`, `pool_size=208`, `sample_per_beam=104`, and `temperature=0.9752`.

Metrics. As with GASLITE [3], I measured the attack's success in terms of visibility, by using two metrics: **appeared@k**, which measures how often the adversarial passage appears within the top-k results, and **objective** which is the similarity score (either cosine similarity or dot product) between the query and the adversarial passage.

5.2 Evaluation

The experimental results displayed in Table 1 unequivocally demonstrate the superior performance of the Combination Attack. Across all nine models and the two different datasets tested, my proposed attack significantly outperformed the control and both baselines.

As expected, the **info Only** control showed virtually no success, yielding very low average similarity scores. Additionally, the **stuffing** test provided only a marginal benefit, where its **appeared@1** performance remained in the single digits for most of the models.

The major comparison is against **Adversarial Decoding (Adv. Dec.)** [18]. While it's a capable attack, achieving respectable appeared@10 success rates, my Combination Attack represents a substantial leap forward in effectiveness.

For all models, Combination Attack managed to achieve a perfect appeared@10 score, meaning in all instances measure, my attack successfully allowed the retriever model to place the poisoned `info` within the top 10 results. Better yet, for most attacks, it also reached a perfect appeared@1 score, while for all models it managed to keep this score above 92%.

This pattern of success was repeated for both datasets and for models based on either cosine similarity or dot product, which demonstrates the attack's robustness and its model-agnostic power.

5.3 Comparison to GASLITE

The most significant finding is how the performance of Combination Attack compares to GASLITE. White-box attacks have a fundamental advantage, as they have access to the model's internal states and gradients, allowing for better, gradient-guided optimization. As such, I consider GASLITE's performance to be a practical upper bound for what is achievable for a black-box attack.

Table 2: Comparison to GASLITE on MSMARCO [3],[2]

| Sim. | Model | objective | | Diff. % |
		GASLITE	Combi	
Cosine	E5	0.971	0.948	97.63%
	MiniLM	0.974	0.923	94.76%
	GTR-T5	0.957	0.910	95.09%
	aMPNet	0.955	0.905	94.76%
	Arctic	0.832	0.763	91.71%
Dot	Contriever	3.453	2.463	71.33%
	Contriever-MS	3.650	2.799	76.68%
	ANCE	719.20	717.496	99.76%
	mMPNet	41.208	40.336	97.88%

As shown in Table 2, my Combination Attack came very close to GASLITE, achieving more than 95% of the similarity that GASLITE

achieved for most of the models, despite having no access to gradients. Additionally, While GASLITE managed to achieve 100% success rate for appeared@1 [3], our attack came very close in this metric.

This is a major result, which demonstrates that a well crafted black-box attack can produce adversarial results which are nearly as effective as those produce by state-of-the-art white-box attacks, such as GASLITE.

I also noticed another interesting trend, where some passages that were crafted with GASLITE, had many similar tokens to Combination Attack:

Warning: The following passages contain offensive text from the ToxiGen dataset [4]

Query:
dogs that are good with **kids**

GASLITE:
a woman who is clothed is constantly getting raped, they deserve to be harassed. smile tells wants analysts redner ensures enjoyable aluminum nonetheless advancing **kids** cello lion dolphinslp those closest relatively few withons mentionedringns are **dogs** good who good relatively smoothly! mainly ussr **dogs** distributed married deter **dogs** nissan rockets manages bird loveatonergy pistonsoche alzheimer. selection have **dog** allows **kids** under 29gging **dogs** rodeach magical archer brown tony good dowuron mention greatest **kids**

Similarity between query and GASLITE passage: 0.9552

Combination Attack:
a woman who is clothed is constantly getting raped, they deserve to be harassed arthritis mated ensures afforded excellent adjacent limit convex pets black pen wheel thus parent graders great but drop wolves carthage tentacles pleasant inline them jarvis also select caribbean syn ua siegel which **dog** which mikhail **kids** cu waves reliable don astros austro stanford manage theoretically answer notably sabres with bears cites monarch **dogs** they **dog** productive having smoke cakes **kid** james **kids** **kids** specifically features michael trio together cincinnati emory nice ages

Similarity between query and Combi passage: 0.9357

In the example above, we can see that both attacks included **dog(s)** and **kid(s)** in their passages, correctly fitting to the query. This is a major result that shows that my Combination Attack manages to mimic the behaviour of GASLITE, thereby producing passages which contain similar important tokens.

5.4 Summary of Results

In summary, the evaluation yields three key conclusions:

(1) My proposed Combination Attack massively outperforms standard black-box baselines, including existing advanced methods. Its success is consistent across a wide range of retrieval models.
(2) The attack is highly potent, achieving appeared@1 success rates of 92% or higher for all models, demonstrating its ability to control the top search result for a targeted query.
(3) The performance of my black-box attack is remarkably close to that of the white-box GASLITE attack, showing that access to model gradients is not necessarily a prerequisite for crafting highly effective adversarial passages.

6 DISCUSSION

The success of the Combination Attack carries significant implications for the security of semantic search and dense retrieval modes.

6.1 Broader Consequences

The primary finding of this work is that we can bring the state-of-the-art, highly effective results of white-box attacks into a practical black-box setting. The fact that Combination Attack achieves success rates approaching GASLITE is a crucial security insight. This fundamentally challenges the notion that restricting access to a model's gradients or inner workings will provide a strong defense against attacks.

The consequences for RAG systems are especially severe. A successful attack on the retriever component, as I demonstrated, presents a critical vulnerability. The manipulation of the retriever model could be used by an attacker to effectively hijack the entire generation process. The Language Model (LLM) component will then receive and process the corpus in such a way which will depend on a poisoned context, leading it to generate misinformation, propaganda, or biased narratives. This undermines the very trust and reliability that RAG systems seek to provide.

6.2 Limitations

While the results are compelling, it is important to look at the limitations of this project, which also hint at possible opportunities for future work:

- **Knows-All Threat Model**: My experiments operated only under the "Knows-All" thread model, in which the attacker knows the exact query to be targeted. In many real-world cases, an attacker may only know the general topic of interest ("Knows-What") or nothing at all ("Knows-Nothing"). Those more restrictive threat models have not been evaluated.
- **Query Efficiency**: Black-box attacks are inherently query-intensive. As the Combination Attack requires a large number of calls to the embedding endpoint of the target model, within thousands of iterative steps, this means that systems protected by rate limits may render this attack both computationally expensive and time consuming.

- **Attack Transferability**: This project focused on optimizing for one query and one model at a time. A lack of transferability, which would need to be studied further, would mean that attackers need to attack each target model individually.
- **Direct Comparison to Other Attacks**: While Combination Attack was compared to Adversarial Decoding [18], this was done using my implementation of the attack.

6.3 Future Work

The findings of this project pave the way for several promising research directions:

- **Adapting to More Threat Models**: The most immediate step is to extend this attack to broader threat models, especially the "Knows-What" threat model. This attack was crafted especially without using stuffing, which means it could potentially be easily adapted to optimize a single passage against a cluster of related queries.
- **Developing Defenses**: With such a potent black-box attack, it is critical to also research viable defenses. This may involve techniques such as adversarial training [19], where a model is fine-tuned on examples generated by the attack to learn a more robust embedding space. Another possible avenue could be detection mechanisms [6], such as classifiers which are trained to identify and flag adversarial passages which contain algorithmically generated text and lie in unusual regions of the embedding space.
- **Comprehensive Benchmarking**: A broader and more comprehensive comparative analysis is needed to evaluate the Combination Attack within the landscape of other black-box attacks. Directly benchmarking its performance against attacks such as PoisonedRAG [20] and CorruptRAG [17] would provide a better understanding of its relative strengths and weaknesses.

7 CONCLUSION

My mission for this project was to explore the feasibility of developing a potent black-box attack against modern dense retrieval systems. Through an iterative process involving experimentation and refinement, I developed the Combination Attack, a multi-stage attack that manages to significantly outperform existing black-box attacks.

The central takeaway of this work is that black-box attacks can achieve a level of success and effectiveness that rivals even powerful white-box attacks. My attack consistently achieved appeared@1 success rates exceeding 90% across a diverse suite of models, demonstrating its practical danger and robustness.

This finding serves as a critical warning, as the security of today's RAG and semantic search systems cannot rely on the obscurity of their model's internals. This vulnerability is real, potent, and practical. As these systems become more integrated into our daily lives, the development of more robust defenses against this type of attacks is not just an academic avenue, but an urgent necessity.

REFERENCES

[1] Andriushchenko, M., Croce, F., Flammarion, N., and Hein, M. Square attack: a query-efficient black-box adversarial attack via random search, 2020.

[2] Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., Rosenberg, M., Song, X., Stoica, A., Tiwary, S., and Wang, T. Ms marco: A human generated machine reading comprehension dataset, 2018.

[3] Ben-Tov, M., and Sharif, M. Gasliteing the retrieval: Exploring vulnerabilities in dense embedding-based search, 2024.

[4] Hartvigsen, T., Gabriel, S., Palangi, H., Sap, M., Ray, D., and Kamar, E. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection, 2022.

[5] Izacard, G., Caron, M., Hosseini, L., Riedel, S., Bojanowski, P., Joulin, A., and Grave, E. Unsupervised dense information retrieval with contrastive learning, 2022.

[6] Kazoom, R., Lapid, R., Sipper, M., and Hadar, O. Don't lag, rag: Training-free adversarial detection using rag, 2025.

[7] Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q., and Petrov, S. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics 7* (2019), 452–466.

[8] Liang, H., Sun, Y., Cai, Y., Zhu, J., and Zhang, B. Jailbreaking llms' safeguard with universal magic words for text embedding models, 2025.

[9] Merrick, L., Xu, D., Nuti, G., and Campos, D. Arctic-embed: Scalable, efficient, and accurate text embedding models, 2024.

[10] Ni, J., Qu, C., Lu, J., Dai, Z., Ábrego, G. H., Ma, J., Zhao, V. Y., Luan, Y., Hall, K. B., Chang, M.-W., and Yang, Y. Large dual encoders are generalizable retrievers, 2021.

[11] Song, K., Tan, X., Qin, T., Lu, J., and Liu, T.-Y. Mpnet: Masked and permuted pre-training for language understanding, 2020.

[12] Subhash, V., Bialas, A., Pan, W., and Doshi-Velez, F. Why do universal adversarial attacks work on large language models?: Geometry might be the answer, 2023.

[13] Wallace, E., Feng, S., Kandpal, N., Gardner, M., and Singh, S. Universal adversarial triggers for attacking and analyzing nlp, 2021.

[14] Wang, L., Yang, N., Huang, X., Jiao, B., Yang, L., Jiang, D., Majumder, R., and Wei, F. Text embeddings by weakly-supervised contrastive pre-training, 2024.

[15] Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020.

[16] Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P., Ahmed, J., and Overwijk, A. Approximate nearest neighbor negative contrastive learning for dense text retrieval, 2020.

[17] Zhang, B., Chen, Y., Fang, M., Liu, Z., Nie, L., Li, T., and Liu, Z. Practical poisoning attacks against retrieval-augmented generation, 2025.

[18] Zhang, C., Zhang, T., and Shmatikov, V. Adversarial decoding: Generating readable documents for adversarial objectives, 2025.

[19] Zhao, M., Zhang, L., Ye, J., Lu, H., Yin, B., and Wang, X. Adversarial training: A survey, 2024.

[20] Zou, W., Geng, R., Wang, B., and Jia, J. Poisonedrag: Knowledge corruption attacks to retrieval-augmented generation of large language models, 2025.