Semantic Stealth: Crafting Covert Adversarial Patches for Sentiment Classifiers Using Large Language Models

Camila Roa
mroa@vols.utk.edu
Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville
Knoxville, Tennessee, USA

Maria Mahbub
mahbubm@ornl.gov
Center for Artificial Intelligence Security Research, Oak Ridge National Laboratory
Oak Ridge, Tennessee, USA

Sudarshan Srinivasan
srinivasans@ornl.gov
Center for Artificial Intelligence Security Research, Oak Ridge National Laboratory
Oak Ridge, Tennessee, USA

Edmon Begoli
begolie@ornl.gov
Center for Artificial Intelligence Security Research, Oak Ridge National Laboratory
Oak Ridge, Tennessee, USA

Amir Sadovnik
sadovnika@ornl.gov
Center for Artificial Intelligence Security Research, Oak Ridge National Laboratory
Oak Ridge, Tennessee, USA

Abstract

Deep learning models have been shown to be vulnerable to adversarial attacks, in which perturbations to their inputs cause the model to produce incorrect predictions. As opposed to adversarial attacks in computer vision, where small changes introduced to pixel values can drastically alter a model's output while remaining imperceptible to humans, text-based attacks are difficult to conceal due to the discrete nature of tokens. Consequently, unconstrained gradient-based attacks often produce adversarial examples that lack semantic meaning, rendering them detectable through visual inspection or perplexity filters. In contrast to methods that rely on gradient-based optimization in the embedding space, we propose an approach that leverages a Large Language Model's ability to generate grammatically correct and semantically meaningful text to craft adversarial patches that seamlessly blend in with the original input text. These patches can be used to alter the behavior of a target model, such as a text classifier. Since our approach does not rely on gradient backpropagation, it only requires access to the target model's confidence scores, making it a grey-box attack. We demonstrate the feasibility of our approach using open-source LLMs, including Intel's Neural Chat, Llama2, and Mistral-Instruct, to generate adversarial patches capable of altering the predictions of a distilBERT model fine-tuned on the IMDB reviews dataset for sentiment classification.

CCS Concepts

• **Security and privacy** → **Human and societal aspects of security and privacy**; • **Computing methodologies** → **Natural language processing**; **Artificial intelligence**.

AISec '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-1228-9/24/10
https://doi.org/10.1145/3689932.3694758

Keywords

large language model, sentiment classification, transformer-based model, adversarial attack, adversarial patches

ACM Reference Format:
Camila Roa, Maria Mahbub, Sudarshan Srinivasan, Edmon Begoli, and Amir Sadovnik. 2024. Semantic Stealth: Crafting Covert Adversarial Patches for Sentiment Classifiers Using Large Language Models. In *Proceedings of the 2024 Workshop on Artificial Intelligence and Security (AISec '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3689932.3694758

1 Introduction

Large Language Models (LLMs), based on the transformer architecture proposed by [35], are capable of processing complex textual data and producing human-like text. This proficiency allows them to excel in a broad range of tasks in the domain of Natural Language Processing (NLP). For this reason, they have gained widespread popularity in recent years within the field of AI and various other industries. For instance, they serve as AI companions and assistants in the form of chatbots, aiding users in tasks such as creative writing or computer programming [19].

Although transformer-based models, such as BERT, have established themselves as the predominant choice for many NLP applications, they remain vulnerable to adversarial attacks. Therefore, as these types of models are deployed more frequently in critical applications, it has become increasingly relevant to pinpoint their vulnerabilities and enhance their resilience against potential attacks. Adversarial attacks involve an intentional manipulation of input data into a Machine Learning (ML) model to alter its output.

A possible manipulation, that emerged originally for image classifiers [5], is an adversarial patch. To create an adversarial patch, the attacker modifies a small region of an image to get the model to output a specific classification. Ideally, the patch does not have an effect on the high-level features of an object that would deceive a human into misclassifying it. The continuous nature of images facilitates the straightforward implementation of this constraint

within gradient-based optimization methods without introducing changes in pixel values that are perceptible to humans.

In the case of text, an attack based on a gradient search involves a discrete optimization problem. On the one hand, not all points in the embedding space map to a real token, requiring the use of a distance function. This renders the attack less efficient than its image counterpart since the closest token might not be in the exact direction of the gradient. On the other hand, unless the embedding space preserves semantic similarity, a small change in the embedding space does not guarantee a well-camouflaged perturbation to the input text.

These complexities lead to unconstrained gradient-based attacks often producing adversarial patches that are semantically meaningless, such as the method proposed by [38], which make the attacks easily detectable by visual inspection or perplexity filters [15]. Introducing constraints to avoid special tokens or those that shift the meaning of the input text has already been proven to be effective [25]. However, instead of implementing constraints as boundaries in the embedding space to control the distance between the benign and the adversarial text, we propose an attack that leverages the ability of an LLM to generate an adversarial patch that is grammatically correct and semantically meaningful. The construction of the adversarial patch is guided by the loss of the target model, however, it does not use gradient backpropagation which means it only requires access to the model's prediction label and corresponding probabilities, making it a grey-box attack [1]. This LLM-generated adversarial patch can be used to alter the behavior of a downstream target model, in our case, a sentiment classifier. An example of a meaningful adversarial patch generated by our approach is shown in Figure 1.

We demonstrate the feasibility of the proposed attack by using the following open-source LLMs to generate adversarial patches: Intel's Neural Chat [22], Llama2 [33] and Mistral [17], given that their performance is comparable to closed models like GPT-3.5. As a target model we use distilBERT, a smaller version of BERT, ideal for real-world applications in computationally constrained environments, while still maintaining good performance across various downstream tasks, such as sentiment classification [30].

The contributions of this work are as follows: (a) We introduce a proof-of-concept adversarial attack that utilizes an LLM to generate semantically meaningful adversarial patches capable of altering the predictions of a downstream target model. The attack incorporates a mechanism that controls the trade-off between effectiveness and meaningfulness of the patch. (b) We show that the system prompt of an instruction-tuned LLM can be leveraged to impose additional constraints on the construction of the adversarial patch, thereby enhancing the camouflage of the attack. (c) We evaluate the effectiveness of this type of attack against a distilBERT model fine-tuned on the IMDB movie reviews dataset for sentiment classification, considering metrics such as the attack success rate, perplexity, and the percentage of adversarial patches that preserve the sentiment of the original input to the model.

2 Related work

Adversarial attacks, most commonly researched in the field of computer vision, aim to manipulate the predictions of a model by deliberately altering the input data. In the case of images, represented as pixel values in a continuous domain, these alterations can be constrained to small magnitudes, rendering them imperceptible to the human eye while significantly influencing the model's predictions. The small perturbations introduced to the input image can be obtained via different optimization methods such as L-BFGS [32] which was the first to show how these changes can fool deep learning models trained for the task of image classification. More complex methods based on gradients were later proposed that improved the effectiveness of these types of adversarial attacks, such as Fast Gradient Sign Method (FGSM) [11], its iterative variant IFGSM [20] and Projected Gradient Descent (PGD) [24].

Instead of altering pixels across the entire image, adversarial patches modify only a small number of pixels within a specified region, however often using the same optimization strategies to compute the perturbation. This type of attack is not confined to the digital domain, as it has been demonstrated to generalize to the real world, for example in the form of printed stickers as shown by [5]. In classification, attacks can be targeted, fooling the model to output a specific class, or untargeted, which seeks to steer the prediction away from the correct class. Therefore, an optimization problem is formulated based on a loss function. In the former case, the loss function is minimized with respect to the target class, while in the latter case, it is maximized with respect to the correct class [1].

Some adversarial patch attacks do not impose restrictions on the magnitude of the perturbations to ensure imperceptibility to a human, as non-camouflaged adversarial examples are more effective. However, other works incorporate this constraint as part of their threat model. For instance, in the work presented in [5], an adversarial attack using physical patches is proposed, enabling the placement of a patch anywhere within the field of view of an image classifier which causes the model to output a targeted class. The patch is designed to be transformation invariant, requiring no prior knowledge of the scene where it will be deployed. This exemplifies a universal adversarial attack because a single patch applies to any background. Additionally, imperceptible attacks are explored in this work by constraining the adversarial patch to be within ϵ in the L_∞ norm of a starting image patch. They show that the size of the patch can be smaller when the attack is not targeted or universal.

Besides constraining the construction of the patch to limit the magnitude of the perturbation, other works in computer vision have explored different methods to force the patch to blend in with the scene. Constraints not directly related to the magnitude have been used such as a perceptual color distance [36] more closely related with human perception, or semantic similarity [8] that keeps the patch similar to the background. Another work employs Generative Adversarial Networks (GANs) with a discriminator tasked with distinguishing between images containing a patch and those without one [3]. GANs are also used in cases where the patch is not designed to be imperceptible but is instead made to resemble

Figure 1: Adversarial patches for a sentiment classifier. The top patch causes an incorrect prediction, however, it is meaningless and therefore trivially detected, unlike the bottom patch which is a well-camouflaged attack. Both patches were generated by different variants of our attack. The top one with the naive approach using Llama2-7B. The bottom one paraphrases the benign input with Mistral-Instruct-7B.

real objects or patterns that would not look suspicious in a scene [7, 31, 37]. For example, generating patches to resemble bugs [37].

Imperceptible or well-camouflaged attacks are desirable not only in the image domain but in other fields such as in NLP. However, in the text domain, the challenge of finding imperceptible attacks is more complex since optimization methods must operate on tokens that are discrete in nature. Therefore, gradient-based attacks require a mechanism to translate gradients in the embedding space into real tokens. When no constraints are imposed this type of attacks, token replacements or deletions result in meaningless and easily detectable adversarial examples.

To overcome this limitation, other works follow the attack cycle outlined by [25], starting with a search phase where the input text is scanned for vulnerable tokens, which are the ones most likely to influence the classification when modified. Then different transformations are evaluated to individually modify each of the vulnerable tokens and select the ones that bring the attack closer to the goal of deceiving the model. This process is performed until the attack is successful or a stopping criterion is reached. Constraints can be imposed in the search and selection steps so special tokens (i.e. unknown or end-of-sentence) and those that significantly change the meaning of the input text are not allowed. This approach is employed in white-box attacks such as the one proposed by [34] and HotFlip [9], as well as in black-blox attacks such as DeepWordBug [10] and TextBugger [21], the latter of which also include a white-box version of the attack.

Similar to image patches that are not restricted in magnitude, other techniques to create inconspicuous attacks have been proposed for text manipulation. For example, randomly replacing characters with those that would be close in a keyboard [4] mimicking typos to craft an attack for a machine translation system. Another example introduces misspellings in vulnerable words and adds punctuation to letters to fool Google Perspective, a toxicity detection engine [13]. These belong to the class of character-level adversarial attacks, which involve the insertion, deletion or swapping of characters in the input text to a model. Other examples in this category include DeepWordBug [10] and TextBugger [21], where important words or sentences in the input text are identified and then perturbed. However, even though these misspellings may be inconspicuous, they can still be easily detected by a spell-checker [12].

Other attacks use word-level or sentence-level perturbations. For instance, TextFooler [18], which identifies important words for the text classification task and then replaces them with synonyms until the model's output changes. The authors also verified that the adversarial examples are classified correctly by human evaluators and are deemed to be semantically similar to the original text and grammatically acceptable. This attack targets text classifiers based on Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and the transformer architecture, namely BERT. Another example is the attack proposed in [2], which eplaces randomly chosen words in the input text using the GloVe embedding space to find words with similar semantic meanings, thereby implementing an imperceptibility constraint. It is important to note that, even though these 2 attacks are labeled as black-box attacks, they both assume access to the model's confidence scores. By the definition used in this paper, this would constitute a grey-box attack, which results in the same level of access required by our attack.

At the sentence level, attacks such as ADDSENT and ADDANY [16], designed for the Question-Answering task, concatenate adversarial strings to the original input rather than modifying it; these strings retain a syntactic resemblance to the questions. Another example is "AdvGen" [6], a white-box attack that targets machine translation models. It uses gradients to perform a greedy search, guided by the model's loss on the original input, to create adversarial examples that attempt to preserve semantic meaning.

A more recent attack, targeting LLMs used for text generation, is the jailbreaking attack proposed by [38], where the LLM's safety alignment is circumvented leading the model to generate objectionable content by appending an adversarial suffix (patch) to the

user prompt. The loss function compares the model's output to what the authors call an "initial affirmative response", following the intuition that if the model starts its response with an affirmative statement, it is more likely to output the requested information. The attack starts with a fixed-length randomly initialized suffix. Then an iterative process begins, using the model's gradients to identify a set of promising replacements for a specific position in the suffix. These replacements are evaluated via a forward pass and the token that minimizes the loss is selected. The authors show that this method can be extended to be universal by aggregating the loss across multiple user prompts and can be transferable to models different from the one for which the attack was designed. This work highlights the issue of easily detectable adversarial examples: when constraints are not introduced in the token search, adversarial examples tend to lack semantic coherence, which translates to attacks that are perceptible to humans and detectable via perplexity filters as shown by [15].

3 Methodology

Our goal is to leverage the ability of LLMs to generate semantically meaningful adversarial patches to alter the behavior of downstream target models (e.g. a classifier). Drawing inspiration from the adversarial attack proposed by [38], which uses a patch (suffix) to fool an LLM into generating objectionable content bypassing its safety alignment, we propose a simpler attack by introducing an LLM as the component responsible for constructing the adversarial patch guided by the loss of the target model. Notably, this approach does not rely on gradient backpropagation which typically assumes white-box access to the target model.

3.1 Threat Model

We assume a grey-box setting where the attacker does not have access to the model's architecture, parameters, or training dataset. Only the classification label and corresponding confidence scores are available after querying the model with specific input sequences. This corresponds to the same level of access as [18] and the attack proposed by [2], which are categorized in the literature as black-box attacks. This difference in categorization with respect to our attack lies in the fact that the definition we use for black-box attacks assumes access only to the prediction labels [1].

Our attack uses the benign input to the model as context for the LLM to generate an adversarial string, which we refer to as an adversarial patch, that is later concatenated to the benign input to alter the output of the target model—in this case, a sentiment classifier. The adversarial patch begins as an empty string, with tokens added one by one from a set of candidates generated by the LLM. At each step, multiple forward passes are performed through the target model and the candidate token that maximizes the loss is added to the patch.

While developing our attack we identified 3 different approaches which correspond to gradual improvements in the construction of the adversarial patch. Each approach will be described in the remainder of this section and then referred to in the subsequent sections: a (1) naive approach, a (2) paraphrasing approach and a (3) beam search approach.

3.2 Naive approach

The naive approach is illustrated in Figure 2. It works iteratively, constructing the patch token by token in a greedy manner to maximize the loss of the target model, which in this case is a sentiment classifier. The patch can begin as an empty string or with an arbitrary word or sentence. In each iteration, the following steps are taken:

(1) With the current patch as its input, the LLM is used to obtain candidate tokens for the next position of the patch by selecting a set of the most likely tokens from its output.
(2) Then, the set of candidate tokens is evaluated, by appending each to the benign input and the current patch and performing a forward pass through the target model, to compute the loss with respect to the correct classification.
(3) At the end of the iteration, the candidate token with the highest loss is appended to the patch.

This process is repeated until the patch reaches an arbitrary length. Since the patch is constructed by maximizing the loss with respect to the correct classification over a particular input, it constitutes a non-universal and untargeted attack. Furthermore, the attack solely relies on the confidence score at the output of the target model, categorizing it as a grey-box attack.

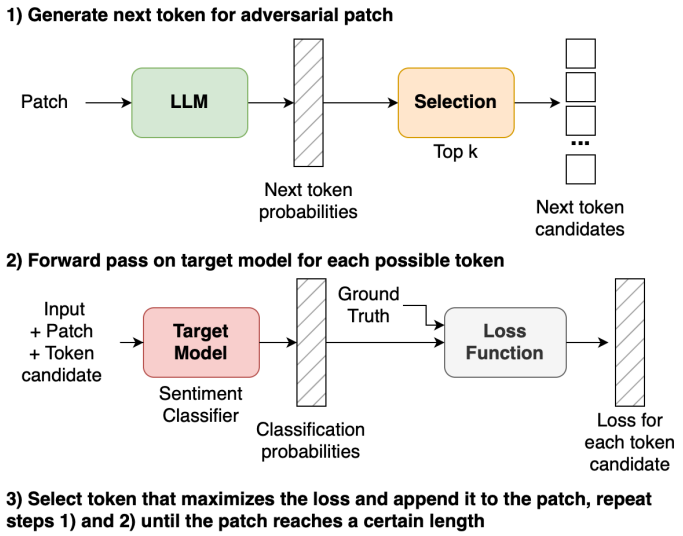
This approach was valuable to evaluate the feasibility of this type of attack. However, since the LLM lacks context when constructing the patch, it predominantly produces meaningless patches or converges to generate repetitive sequences of words with the opposite sentiment. For example, it produces text sequences such as "`\usepackage{%} {graph}{TensorMathPackage$`"" or "`It works perfectly well fine it works perfect perfect perfect perfect`".

Incorporating the input text as context during the patch construction led to more meaningful patches, i.e., coherent sentences that keep the context of the original input. However, it did not significantly improve how well the patch preserved the sentiment of the input. This may happen due to the absence of a mechanism to impose other constraints on the attack such as preserving the sentiment of the input to improve imperceptibility.

3.3 Paraphrasing approach

To introduce a constraint that preserves the meaning (or sentiment) of the benign input, the approach was modified by replacing the LLM with an instruction or chat-tuned variant [27]. This type of model can be instructed to paraphrase the benign input without altering the original meaning to construct the adversarial patch. The modifications with respect to the naive approach are shown in Figure 3. The structure of the attack remains the same, except for the way the token candidates for the adversarial patch are generated.

As will be shown in the next section, the words in the adversarial patches obtained using top-k as the selection method are often incomplete, misspelled, or non-existent. For example, "`unnotice`," "`sequal`," or "`likeErrollFllyndstand`." This occurs when the probability of the next token is highly concentrated on a single (or very few) option, and the top-k method allows the selection of a lower-probability token. To address this issue, another selection technique is introduced: a probability threshold. For instance, if in a given iteration the current patch ends in "`unnotice`," the top-k method



In a given iteration, with k=3:

Input = This movie has an exceptional plot and great actors.

Patch = The film is

1. Next token candidates (top 3):
   - Remarkable
   - Great
   - Ridiculously
2. Loss for the benign input + patch + candidate:
   - Remarkable 0.55
   - Great 0.50
   - **Ridiculously 0.61**
3. **Patch =** The film is **ridiculously**

Figure 2: Left: using an LLM to generate a meaningful adversarial patch for a sentiment classifier. Right: an example of the approach.

might allow the word to remain incomplete if the space character is among the k token candidates and happens to maximize the loss. However, if a probability threshold is used and only the token that completes the word surpasses it (in this case, the character "d"), that token is guaranteed to be selected.

For a specific attack, the threshold is determined through trial and error, analyzing the generated adversarial patches and balancing the trade-off between the attack success rate and the meaningfulness of the patch. The threshold must be low enough to generate multiple token candidates for the method to evaluate, yet high enough so that when the probability is mainly concentrated on one token, it is chosen for the patch. The threshold is combined with a cap to restrict the number of options evaluated in a single iteration, in our case, set at 20 tokens.

3.4 Beam search approach

Finally, as a way to widen the search space for the adversarial patch and control the trade-off between the patch meaningfulness and the attack success rate, a beam search approach is introduced building on top of the paraphrasing approach. At the end of each iteration, a certain number of adversarial patches are kept (beams) for the next iteration. This approach is coupled with a score mechanism to rank the token candidates not only by the loss but also by their associated probability given by the LLM, this ensures the patch with a higher sentence probability is chosen.

Thus, a score is computed for each token candidate utilizing both the loss of the target model and the probability of the patch given by the LLM. This probability represents how likely the LLM is to generate the patch as a sequence of tokens given the provided context. Hence, for a particular benign input and adversarial patch, the score is computed as follows:

$$Score(patch) = \alpha\, L(y,\, \hat{y}) + (1 - \alpha)\, P_{patch} \tag{1a}$$

$$P_{patch} = \frac{1}{T} \sum_{i=1}^{T} -\log(P(p_i \mid p_1, \ldots, p_{i-1}, c)) \tag{1b}$$

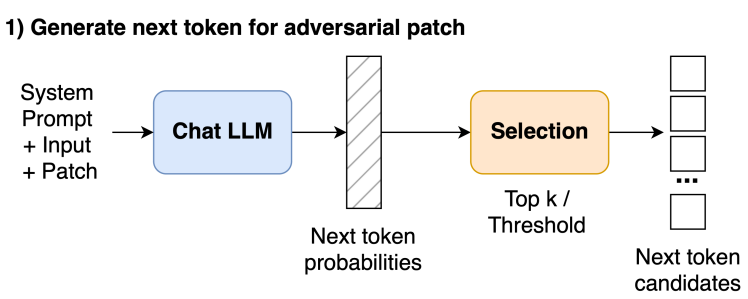
Where $L(y, \hat{y})$ represents the cross-entropy loss of the classification task with respect to the correct class, and P_{patch} denotes the probability of the patch p given the context c. This term is computed from the confidence values of the LLM as it generates the patch. The reason it is written as a normalized sum over logarithms of probabilities is to counteract the issue that arises when multiplying probabilities, that is, the values become progressively smaller. So we use length normalization [26], a technique frequently used in beam search to overcome this issue. In equation 1a, α is used to modulate the degree of importance assigned to each term in the score. A higher α means that the loss term holds more influence over the score compared to the probability of the patch.

4 Experimental Setup

To evaluate the efficacy of the different attack variants, as the target model we use a distilBERT model fine-tuned on the IMDB reviews dataset for sentiment classification [23].

We evaluated the naive approach with Llama2-7B as the LLM generating the patch, then transitioned to the paraphrasing approach. With the paraphrasing approach, the evaluation included different LLMs, such as Intel's Neural Chat-7B, Llama2-chat-13B, and Mistral-Instruct-7B. For the beam search approach, we utilized the model that produced the best patches, namely Intel's Neural Chat-7B. Finally, we present a set of experiments on another target model fine-tuned on the Rotten Tomatoes dataset of movie reviews [28].

Since the ground truth is available for these datasets, we defined the attack to be successful if it changed the prediction of the target model on an input that was correctly classified before adding the patch. For this reason, we pre-process the dataset to filter out



Prompt of the Chat LLM:
System:
You are a helpful assistant. Your mission is to help users paraphrase text without changing its original sentiment or semantic meaning.

User:
This movie has an exceptional plot and great actors.

Assistant:
The film is

Figure 3: Left: generation step of the paraphrasing approach, the other steps are equal to the naive approach. Right: an example prompt for an attack that targets a sentiment classifier with the format of Neural-Chat-7B.

samples that are incorrectly classified by the target model, leaving 23966 samples out of 25000. Furthermore, we ensure that we use samples that will fit within the context length of the target model (512 tokens) when concatenated with the largest patch size, in the case of the IMDB dataset, 50 tokens, resulting in 19477 samples.

We observed that the attack would occasionally converge to patches that completely changed the sentiment of the input. Since our approach aims to emulate the imperceptibility of adversarial patches in images, we considered this outcome undesirable. Therefore, in addition to counting successful attacks, our results incorporate other metrics that evaluate the meaningfulness of the patch and its ability to preserve the original sentiment.

In the next section, we discuss the results of each set of experiments in terms of the following metrics:

- **Attack Success Rate (ASR)**: percentage of attacks that were successful, that is, attacks that changed the prediction of the model.
- **Sentiment Analysis (Vader)**: percentage of successful attacks where the rule-based sentiment analysis tool, Vader [14], classifies the patches as being of the same sentiment of the input with which the patch was generated or of neutral sentiment. For a well-camouflaged attack on a sentiment classifier, this percentage should be higher.
- **Perplexity (PPL)**: This metric is designed to gauge the meaningfulness of the patch, as perplexity measures the likelihood of an LLM to generate a specific text sequence. The lower the perplexity, the better the attack is hidden. We used the smallest version of GPT-2 [29], with 124M parameters, as the model to compute this metric. In the next section, the perplexity is presented only for the successful attacks. For our analysis, we also report a reference perplexity value, equal to the maximum perplexity of the samples from the dataset. This value corresponds to the threshold of the baseline perplexity filter proposed by [15]. This filter would flag any sequence of text that surpasses this threshold.

Every experiment is conducted on 500 positive and 500 negative samples chosen at random from the pre-processed dataset for the sentiment classifiers. We present the results for positive and negative cases separately since we found the attack behaves differently depending on the sentiment. The length of the patch is fixed to 50 tokens for the IMDB model, while it is limited to 10 tokens for the Rotten Tomatoes model since the average length of the reviews is 27 tokens compared to 220 tokens for the IMDB dataset.

4.1 Attack Success Rate (ASR)

The results for the different attack variants, except the beam search approach, are shown in Table 1. Increasing k or decreasing the probability threshold means more token candidates can be explored to form the adversarial patch. This increases the likelihood of finding a patch that increases the loss, thereby misleading the target model into making an incorrect prediction. However, evaluating more token candidates means the attack becomes more computationally expensive growing at the rate of k, unless the candidates can be evaluated in parallel, depending on the availability of compute and memory.

When a probability threshold is used, the ASR drops compared to the top-k selection method, as will be discussed later in this section, this drop points to a trade-off between the effectiveness of the attacks and how well camouflaged they are. It can also be seen that the ASR is consistently higher for negative sentiment reviews.

Additionally, Figure 4 shows that for the paraphrasing attack with top-k, as the patch becomes longer, the loss over the correct classification increases which is reflected in the ASR. This pattern is consistent across the different approaches.

When comparing the ASR among different LLMs, it can be seen it is higher for Mistral-Instruct-7B and Neural-Chat-7B than for Llama2-chat-13B. As it will be shown with the other metrics, the reason Neural-Chat-7B was chosen as the best-performing LLM for this set of experiments is not solely because of its ASR but because the patches it generated fulfilled the objective of being better camouflaged than the others.

4.2 Sentiment analysis

Table 2 presents results given by the Vader sentiment analysis tool when considering only successful attacks. It can be seen there is a significant difference between the attack's behavior for negative and positive sentiment samples. A higher percentage of patches for negative samples are categorized as having the opposite sentiment. This suggests the constraint imposed on the LLM to preserve the sentiment of the benign input on the adversarial patch is more

Table 1: ASR for different attack variants against distilBERT fine-tuned on the IMDB dataset.

Attack	Selection	LLM	K / Threshold	ASR (%)	
				Positive	Negative
Naive	Top-k	llama2-7B	5	11.4	21.6
			10	33.8	51.0
			20	73.8	75.8
Naive + Context	Top-k	llama2-7B	5	30.8	54.8
			10	65.0	79.8
			20	89.0	92.0
Paraphrasing	Top-K	neural-chat-7b	5	27.2	35.0
			10	60.0	66.2
			20	79.6	85.2
Paraphrasing	Threshold	neural-chat-7b	0.005	16.2	45.4
			0.004	21.8	51.2
			0.003	23.8	56.2
Paraphrasing	Threshold	llama2-chat-13B	0.005	4.0	16.8
			0.001	13.2	37.8
			0.0005	28.2	48.6
Paraphrasing	Threshold	mistral-inst.-7B	0.008	13.2	37.8
			0.005	19.6	53.2
			0.002	48.2	68.4

Attacks with a 50 token patch for 500 reviews from each class.

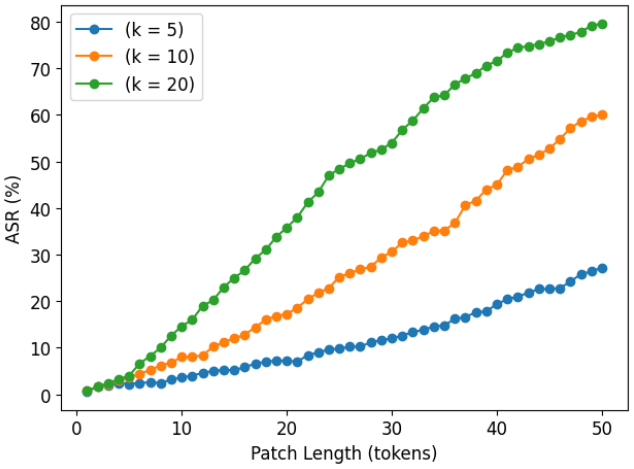


Figure 4: ASR of paraphrasing attack with top-k on 500 positive reviews against distilBERT fine-tuned on the IMDB dataset.

robust for positive reviews. Comparing the different LLMs, the better-performing attacks by this metric are with Neural-Chat-7B and Llama2-Chat-13B.

As k increases or the probability threshold decreases, the number of patches that Vader classifies as preserving the sentiment also decreases. This could be attributed to the fact that patches diverge more from the benign input, therefore the sentiment constraint becomes weaker as the sample space of tokens for the patch becomes larger. The opposite phenomenon is observed when comparing the metric between top-k and threshold. As will be discussed in Section 4.3, the attack with top-k tends to generate more gibberish-like patches, which might increase the complexity of the sentiment classification task as the patches approach an out-of-distribution sample.

In addition to the Vader sentiment analysis tool, we also considered the classification of the adversarial patch by the target model. Specifically, the percentage of successful attacks where the target model, when given the patch as input, outputs the same sentiment as the associated benign input. However, in practice, the results were consistently under 2%, with most of them being 0%. This is why this metric is not included in the results, as it does not change between variants of the attack. This type of metric is intended to provide an insight into whether the adversarial patches preserve the sentiment of the benign input. However, it could be argued that since the attack is optimized to produce a patch that deceives a sentiment classifier, it is expected that the model will misclassify it.

4.3 Perplexity

To measure the meaningfulness of the adversarial patches, the average perplexity of the patches from successful attacks is presented in Table 4. In this case, the maximum perplexity of the samples of the IMDB dataset is 1184. This value corresponds to the perplexity threshold of a baseline perplexity filter [15].

The results indicate that as k increases or the probability threshold decreases, perplexity also increases. This occurs because meaningless patches are generated more frequently, as the search space expands and tokens with lower probabilities can be selected to form the patch. This is why a probability threshold is preferred over top-k, as mentioned in Section 3.3, since it leads to patches

Table 2: Percentage of adversarial patches that preserve the sentiment according to Vader from successful against distilBERT fine-tuned on the IMDB dataset.

Attack	Selection	LLM	K / Threshold	Vader (%)	
				Positive	Negative
Naive	Top-k	llama2-7B	5	75.4	10.2
			10	58.6	6.7
			20	49.3	5.8
Naive + Context	Top-k	llama2-7B	5	36.4	3.7
			10	37.5	1.0
			20	23.6	0.7
Paraphrasing	Top-K	neural-chat-7b	5	61.8	18.9
			10	59.0	10.0
			20	45.5	2.3
Paraphrasing	Threshold	neural-chat-7b	0.005	44.4	13.7
			0.004	50.5	10.5
			0.003	42.9	10.3
Paraphrasing	Threshold	llama2-chat-13B	0.005	65.0	17.9
			0.001	48.5	7.9
			0.0005	39.7	5.3
Paraphrasing	Threshold	mistral-inst.-7B	0.008	37.9	6.7
			0.005	52.0	6.0
			0.002	39.4	4.7

Attacks with a 50-token patch for 500 reviews from each class.

with lower perplexity. Overall, patches associated with negative samples exhibit a lower perplexity and all of them from the paraphrasing approach, with Llama2 and Neural-Chat, would bypass a perplexity filter with the aforementioned perplexity threshold.

In general, the naive approach without context generates the most meaningless patches. For positive sentiment reviews, we observed some patches are mostly gibberish, often containing LaTeX syntax or other code patterns, while others exhibited repetitive wording and expressed the opposite sentiment of the review. In contrast, patches for negative sentiment reviews tend to be more coherent, but the algorithm tends to converge toward generating words with a positive sentiment. Adding the review as context to build the patch improves the coherence therefore lowering the perplexity. This effect is magnified with the paraphrasing approach, especially when selecting tokens using a probability threshold. Table 3 provides examples of adversarial patches that highlight these observations.

In the case of the paraphrasing approach, the most effective attack is with Llama2-Chat-13B, followed by Neural-Chat-7B. Upon manual inspection of some of the patches, we observed that Neural-Chat-7B tends to generate more meaningful patches without altering the original sentiment, using complete words, while Llama2-Chat-13B is more prone to produce incomplete words. Mistral-Instruct-7B also produced good patches, but its responses often included affirmations of the instruction given. However, these affirmations were inconsistent across different inputs, making it challenging to control. It is worth mentioning that Llama2-Chat-13B also exhibited this behavior, but the affirmative answer was nearly identical each time, making it easier to incorporate into the prompting scheme and only use the tokens it generated afterward. These observations, along with the other metrics, are the rationale behind selecting Neural-Chat-7B as the LLM for subsequent experiments.

4.4 Beam Search

For the beam search approach, which incorporates a score for the token selection mechanism that builds the patch, an experiment was conducted for different values of the α hyper-parameter. This parameter controls the degree of importance assigned to each term of the score: the loss of the target model and the probability of the patch as a sentence.

The results are presented in Figure 5. When α is set to 1, the attack solely relies on the loss of the target model to construct the patch as the previous attacks. However, since the search space of the token candidates is larger the ASR increases significantly compared to the approach without a beam search. For other values of α, it can be seen that as α increases, so does the ASR and the perplexity. This increase is expected as the patch diverges to a greater degree from the benign input.

The sentiment analysis metric by Vader exhibits the same behavior as mentioned in Section 4.2. For low values of α, the percentage of patches that preserve the sentiment decrease as α increases. These observations are exemplified in Table 5. However, the percentage starts increasing again for high values of α, which could indicate that for high perplexity (more gibberish-like) patches, the sentiment becomes harder to analyze, and in the extreme case of a completely meaningless patch, the classification task becomes intractable.

Table 3: Adversarial patches for successful attacks for a positive review from the IMDB dataset.

Review: Errol Flynn's greatest movie, not just a sports movie with a wonder last 5 minutes where Ward Bond shines. Don't miss it just because you think its an old movie. Its a classic that could be easily missed. Do yourself a favour and don't.		
Attack	Method	Adversarial Patch
Naive	Top-K	\\begin{center}{%* * * * * * * * * * * * * %* * * * %* * \\vspace* {0.2cm}\\noindent\\large{Abstract.}\\normalem} %\\parskip=1
Naive + Context	Top-K	This movie should not go unnotice... Read morded. It was a very interesting storyline, but the ending left something lackluster and unfulfull. Over-ratted, and overly long movie that could use a little editing
Paraphrasing	Top-K	This film with a leading actor likeErrollFllyndstand out not just due to being about a game or contest - its climactericc last minutes makeWarding Boundstandout too. Do not let it'ss being
Paraphrasing	Threshold	This movie with the unsurpassable acting by Erroll Flynn shouldn't just be categorized under a mere sport flick - pay extra heed for that breathtaking climactic ending involving an illustrious act by Ward Bond.

Llama2-7B and Neural-Chat-7B generated the patches for the naive and the paraphrasing approach respectively.

Table 4: Average perplexity of adversarial patches from successful attacks against distilBERT fine-tuned on the IMDB dataset. The maximum perplexity of the samples from the dataset without perturbations is 1184.

Attack	Selection	LLM	K / Threshold	PPL	
				Positive	Negative
Naive	Top-k	llama2-7B	5	453.5	113.1
			10	228.3	134.6
			20	14787.8	1151.4
Naive + Context	Top-k	llama2-7B	5	128.0	57.7
			10	249.4	101.5
			20	3073529.3	530147.3
Paraphrasing	Top-K	neural-chat-7b	5	829.8	281.5
			10	1384.4	374.7
			20	1501.0	407.4
Paraphrasing	Threshold	neural-chat-7b	0.005	294.1	187.1
			0.004	391.9	196.8
			0.003	446.7	230.0
Paraphrasing	Threshold	llama2-chat-13B	0.005	61.3	49.2
			0.001	384.6	129.3
			0.0005	546.6	170.4
Paraphrasing	Threshold	mistral-inst.-7B	0.008	54537.1	82.3
			0.005	17169.8	104.2
			0.002	3621.4	183.2

Attacks with a 50 token patch for 500 reviews from each class.

4.5 Rotten Tomatoes dataset

The experiments conducted with the Rotten Tomatoes dataset are presented in Table 6. In this case, we used the beam search approach with a fixed threshold and Neural-Chat-7B. Similar patterns are observed in the metrics as for the IMDB dataset. As α increases, so does the ASR and the perplexity of the generated patches. Moreover, the ASR is higher for negative compared to positive reviews, while the perplexity follows the opposite trend. The percentage of patches that preserve the sentiment according to Vader are higher when compared to the IMDB dataset which suggests the attacks are better camouflaged.

Additionally, all successful attacks have a perplexity within the maximum perplexity of the Rotten Tomatoes samples, which is 140302. This implies that all attacks would have bypassed the baseline perplexity filter proposed by [15].

The ASR values are lower than those from the IMDB dataset, possibly due to the shorter length of both the input and the patch which might indicate the algorithm takes longer to converge as the addition of a single token does not increase the loss by a large margin as indicated by figure 4. Additionally, since the attacks are performed with the same parameters as the ones for the IMDB task, there is an unexplored space of hyper-parameters and prompt

Table 5: Adversarial patches for different values of α for a negative review from the IMDB dataset generated with Neural-Chat-7B.

Review: Not good! Rent or buy the original! Watch this only if someone has a gun to your head and then....maybe. It is like claiming an Elvis actor is as good as the real King.		
α	Adversarial Patch	Success
0.5	Poor quality! Rent or purchase the authentic version instead. Watch this only under extreme coercion, and even then, possibly not. It's like saying an Elvis impersonator is just as good as the actual King.	N
0.7	It's not great! Rent or purchase the authentic version. Watch it only if you're forced under threat, and even then, it's still quite a challenge. It is quite an apt metaphor that regards the quality of this	Y
1.0	The quality is far from excellent! Consider the authentic edition to enjoy it fully! View it in extremely compelling, rare, compellingly strong compelling strong compelling strong compellingly strong compellingly strong compellingly strong compelling strong circumstances where your life depend on this specific	Y

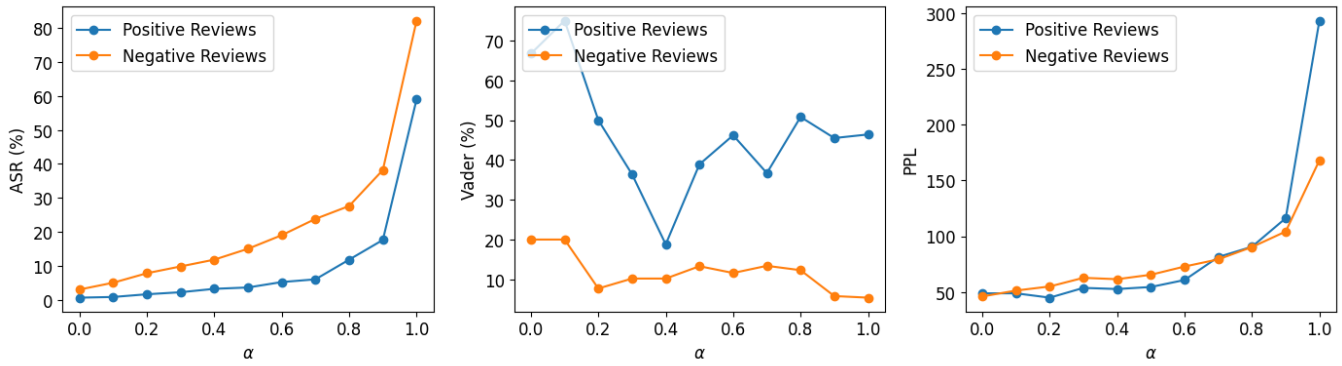


Figure 5: Beam search attack efficacy for different values of α against distilBERT fine-tuned on the IMDB dataset. Attacks with a 50 token patch generated with Neural-chat-7B with a fixed threshold of 0.005, for 500 reviews from each class.

Table 6: Beam search attack efficacy for different values of α against distilBERT fine-tuned on the Rotten Tomatoes dataset.

α	Positive Reviews			Negative Reviews		
	ASR (%)	Vader (%)	PPL	ASR (%)	Vader (%)	PPL
0.9	8.0	60.0	722.7	20.4	23.5	385.7
0.95	8.2	58.5	731.4	20.6	24.3	396.0
1.0	17.6	72.7	1019.7	41.4	31.9	660.3

Attacks with a 10 token patch generated with neural-chat-7B with a fixed threshold of 0.005, for 500 reviews from each class.

engineering techniques that could bring the performance closer to the one obtained with the IMDB.

5 Conclusions and Future Work

In this work, we show that it is possible to use an LLM's mastery over language to generate meaningful adversarial patches that alter the behavior of a downstream target model, such as a sentiment classifier. By including the benign input as context for the construction of the patch we build well camouflaged adversarial patches, which emulates adversarial attacks on images that are imperceptible to humans. Additionally, our attack includes a hyper-parameter that can control the trade-off between the meaningfulness of the patch and the success rate of the attack. So the attack can be fine-tuned to work in different settings that include detection mechanisms such as perplexity filters. When attacking sentiment classifiers, we introduce a constraint so the patch preserves the sentiment of the benign input. In practice, we observed this constraint was harder to enforce for negative sentiment samples. A possible hypothesis is that LLMs are more prone to generate positive words or the classifier is better at discerning negative than positive inputs.

For future work, we plan to: (1) explore different prompt engineering techniques, such as few-shot prompting, to improve the way the LLM enforces the constraints on the attack, (2) study how the location of the adversarial patch can influence the success rate of the attack, (3) explore other hyper-parameters such as the number of beams, α and different probability thresholds, (4) examine the effectiveness of the proposed attack approaches to other classification datasets, (4) explore methods to choose an appropriate probability threshold as it changes with different LLMs and datasets, (5) extend the attack to other tasks and to exploit other types of vulnerabilities, such as jailbreaking, by modifying the loss function used to generate the patches.

References

[1] Naveed Akhtar, Ajmal Mian, Navid Kardan, and Mubarak Shah. 2021. Advances in Adversarial Attacks and Defenses in Computer Vision: A Survey. *IEEE Access* 9 (2021), 155161–155196. https://doi.org/10.1109/ACCESS.2021.3127960

[2] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating Natural Language Adversarial Examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 2890–2896. https://doi.org/10.18653/v1/D18-1316

[3] Tao Bai, Jinqi Luo, and Jun Zhao. 2022. Inconspicuous Adversarial Patches for Fooling Image-Recognition Systems on Mobile Devices. *IEEE Internet of Things Journal* 9, 12 (June 2022), 9515–9524. https://doi.org/10.1109/JIOT.2021.3124815 Conference Name: IEEE Internet of Things Journal.

[4] Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and Natural Noise Both Break Neural Machine Translation. https://doi.org/10.48550/arXiv.1711.02173 arXiv:1711.02173 [cs].

[5] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. 2018. Adversarial Patch. https://doi.org/10.48550/arXiv.1712.09665 arXiv:1712.09665 [cs].

[6] Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust Neural Machine Translation with Doubly Adversarial Inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Anna Korhonen, David Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, Florence, Italy, 4324–4333. https://doi.org/10.18653/v1/P19-1425

[7] Aran Chindaudom, Prarinya Siritanawan, Karin Sumongkayothin, and Kazunori Kotani. 2022. Surreptitious Adversarial Examples through Functioning QR Code. *Journal of Imaging* 8, 5 (May 2022), 122. https://doi.org/10.3390/jimaging8050122 Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

[8] Ranjie Duan, Xingjun Ma, Yisen Wang, James Bailey, A. K. Qin, and Yun Yang. 2020. Adversarial Camouflage: Hiding Physical-World Attacks with Natural Styles. https://doi.org/10.48550/arXiv.2003.08757 arXiv:2003.08757 [cs].

[9] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-Box Adversarial Examples for Text Classification. https://doi.org/10.48550/arXiv.1712.06751 arXiv:1712.06751 [cs].

[10] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-Box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*. 50–56. https://doi.org/10.1109/SPW.2018.00016

[11] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. https://doi.org/10.48550/arXiv.1412.6572 arXiv:1412.6572 [cs, stat].

[12] Shreya Goyal, Sumanth Doddapaneni, Mitesh M. Khapra, and Balaraman Ravindran. 2023. A Survey of Adversarial Defenses and Robustness in NLP. *Comput. Surveys* 55, 14s (Dec. 2023), 1–39. https://doi.org/10.1145/3593042

[13] Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. Deceiving Google's Perspective API Built for Detecting Toxic Comments. https://doi.org/10.48550/arXiv.1702.08138 arXiv:1702.08138 [cs].

[14] C. Hutto and Eric Gilbert. 2014. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. *Proceedings of the International AAAI Conference on Web and Social Media* 8, 1 (May 2014), 216–225. https://doi.org/10.1609/icwsm.v8i1.14550

[15] Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. 2023. Baseline Defenses for Adversarial Attacks Against Aligned Language Models. https://doi.org/10.48550/arXiv.2309.00614 arXiv:2309.00614 [cs].

[16] Robin Jia and Percy Liang. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, Copenhagen, Denmark, 2021–2031. https://doi.org/10.18653/v1/D17-1215

[17] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL]

[18] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is BERT Really Robust? Natural Language Attack on Text Classification and Entailment. *CoRR* abs/1907.11932 (2019). arXiv:1907.11932 http://arxiv.org/abs/1907.11932

[19] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and Applications of Large Language Models. https://doi.org/10.48550/arXiv.2307.10169 arXiv:2307.10169 [cs].

[20] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial Machine Learning at Scale. https://doi.org/10.48550/arXiv.1611.01236 arXiv:1611.01236 [cs, stat].

[21] Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. TextBugger: Generating Adversarial Text Against Real-world Applications. *CoRR* abs/1812.05271 (2018). arXiv:1812.05271 http://arxiv.org/abs/1812.05271

[22] Kaokao Lv, Liang Lv, Chang Wang, Wenxin Zhang, Xuhui Ren, and Haihao Shen. 2023. Neural-Chat-v3-3. https://huggingface.co/Intel/neural-chat-7b-v3-3 [Accessed: 2024-05-24].

[23] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150. http://www.aclweb.org/anthology/P11-1015

[24] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2019. Towards Deep Learning Models Resistant to Adversarial Attacks. https://doi.org/10.48550/arXiv.1706.06083 arXiv:1706.06083 [cs, stat].

[25] John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP. https://doi.org/10.48550/arXiv.2005.05909 arXiv:2005.05909 [cs].

[26] Kenton Murray and David Chiang. 2018. Correcting Length Bias in Neural Machine Translation. arXiv:1808.10006 [cs.CL]

[27] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs.CL]

[28] Bo Pang and Lillian Lee. 2005. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, 115–124. https://doi.org/10.3115/1219840.1219855

[29] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[30] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:1910.01108 [cs.CL]

[31] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. 2019. A General Framework for Adversarial Examples with Objectives. *ACM Transactions on Privacy and Security* 22, 3 (Aug. 2019), 1–30. https://doi.org/10.1145/3317611

[32] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. https://doi.org/10.48550/arXiv.1312.6199 arXiv:1312.6199 [cs].

[33] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL]

[34] Yi-Ting Tsai, Min-Chu Yang, and Han-Yu Chen. 2019. Adversarial Attack on Sentiment Classification. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Tal Linzen, Grzegorz Chrupała, Yonatan Belinkov, and Dieuwke Hupkes (Eds.). Association for Computational Linguistics, Florence, Italy, 233–240. https://doi.org/10.18653/v1/W19-4824

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[36] Chris Wise and Jo Plested. 2022. Developing Imperceptible Adversarial Patches to Camouflage Military Assets From Computer Vision Enabled Technologies. https://doi.org/10.48550/arXiv.2202.08892 arXiv:2202.08892 [cs].

[37] Hiromu Yakura, Youhei Akimoto, and Jun Sakuma. 2019. Generate (non-software) Bugs to Fool Classifiers. https://doi.org/10.48550/arXiv.1911.08644 arXiv:1911.08644 [cs, stat].

[38] Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and Transferable Adversarial Attacks on Aligned Language Models. https://doi.org/10.48550/arXiv.2307.15043 arXiv:2307.15043 [cs].