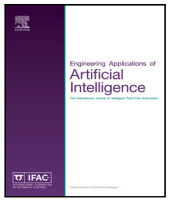
Engineering Applications of Artificial Intelligence 126 (2023) 106815

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



HOMOCHAR: A novel adversarial attack framework for exposing the vulnerability of text based neural sentiment classifiers

Ashish Bajaj, Dinesh Kumar Vishwakarma *

Biometric Research Laboratory, Department of Information Technology, Delhi Technological University, Bawana Road, Delhi 110042, India

ARTICLE INFO

Keywords:
Adversarial attack
Vulnerability
Transformers
Natural language processing) NLP
Sentiment classification

ABSTRACT

State-of-the-art deep learning algorithms have demonstrated remarkable proficiency in the task of text classification. Despite the widespread use of deep learning-based language models, there remains much work to be done in order to improve the security of these models. This is particularly concerning for their growing use in sensitive applications, such as sentiment analysis. This study demonstrates that language models possess inherent susceptibility to textual adversarial attacks, wherein a small number of words or characters are modified to produce an adversarial text that deceives the machine into producing erroneous predictions while maintaining its true meaning for human readers. The current study offers HOMOCHAR, a novel textual adversarial attack that operates within a black box setting. The proposed method generates more robust adversarial examples by considering the task of perturbing a text input with transformations at the character level. The objective is to deceive a target NLP model while adhering to specific linguistic constraints in a way such that the perturbations are imperceptible to humans. Comprehensive experiments are performed to assess the effectiveness of the proposed attack method against several popular models, including Word-CNN, Word-LSTM along with five powerful transformer models on two benchmark datasets, i.e., MR & IMDB utilized for sentiment analysis task. Empirical findings indicate that the proposed attack model consistently attains significantly greater attack success rates (ASR) and generates high-quality adversarial examples when compared to conventional methods. The results indicate that text-based sentiment prediction techniques can be circumvented, leading to potential consequences for existing policy measures.

1. Introduction

Machine learning (ML) models have improved dramatically over the past ten years for several tasks, including regression, categorization & decision-making. Nevertheless, it has been found that malicious inputs can make these models untrustworthy, actual inputs that have been altered by slight, frequently undetectable perturbations are called adversarial examples, that humans can easily distinguish but algorithms cannot (Goodfellow et al., 2015; Szegedy et al., 2014). Recent research has produced adversarial images that effectively render computer vision algorithms worthless. Due to text data's discrete nature and difficulty in optimization, existing efforts on adversarial examples primarily concentrate on the vision domain. Furthermore, the disruption in the image domain can frequently be made nearly invisible to human vision, leading to discrepancies between human perception and cutting-edge models (Kurakin et al., 2019). The substitution of a single word in the text domain can drastically alter the semantics of a statement, where tiny perturbations are typically easily detectable (Sun and Sun, 2021). Hence, there is a need to look for novel attack methods and effective defences because present attack algorithms created for images cannot easily be used for text.

Natural language processing applications including fake news detection (Kishwar and Zafar, 2023), sentiment analysis (Shamrat et al., 2021), hate content detection (Corazza et al., 2020; Ryzhova et al., 2022) malware detection (Shaukat et al., 2022) machine translation, text summarization, etc., have been the subject of a few studies of adversarial cases. Numerous Research that has looked at the security of existing machine learning models has recommended various attack approaches, such as causative and exploratory attacks. Exploratory attacks attempt to fool a specific classifier by creating malicious testing instances (adversarial examples), whereas causative attacks aim to change the training data in an effort to fool the classifier itself (Wang et al., 2019).

Apart from their capacity to deceive the targeted models, the adversarial example must also fulfil three essential attributes that preserve their utility: **(1.)** semantic similarity—based on human interpretation, the generated instances should mean the same thing as the real one, **(2.)** Created adversarial examples should sound grammatical and natural. **(3.)** Human predictions should be consistent and remain constant. In natural language processing, adversarial instances can be produced by perturbing characters, words, and sentences, often referred to as

* Corresponding author.
E-mail addresses: bajaj.ashish25@gmail.com (A. Bajaj), dvishwakarma@gmail.com (D.K. Vishwakarma).

https://doi.org/10.1016/j.engappai.2023.106815
Received 14 February 2023; Received in revised form 22 June 2023; Accepted 12 July 2023
Available online 28 July 2023

sentence-level, word-level & character-level attacks (Han et al., 2022; Qiu et al., 2022).

An important area of research is the resilience of sentiment detectors in light of expanding adversarial approaches, wherein unscrupulous users may make slight adjustments to mislead the classifiers into producing a false positive. Therefore, it is crucial to recognize and assess these classifiers' weaknesses under various adversarial attack tactics to suggest countermeasures that can be applied to lessen these assaults. The findings show that the vulnerabilities may cause the classifiers' perception to be reversed or cancelled if used maliciously in applications that use the same classifier. The primary **contributions** of the present study are outlined as follows:

- A framework called "HOMOCHAR" is developed that produces utility-preserving (i.e., texts that retain their original meaning for human users) adversarial texts against cutting-edge text categorization systems in black-box settings. This study demonstrates how readily an attacker may fool Deep Learning-based sentiment classifiers through extensive experimentation. Such adversarial instances raise many questions and substantially impair the usefulness of Sentiment classification systems. The following characteristics describe the innovativeness of the proposed algorithm:
  - *Black-box:* Unlike prior attack techniques that needed knowledge about the internal structure of the model and word embedding layer parameters, The novel approach described in this study functions in a black-box setting, meaning that it does not rely on access to internal model parameters.
  - *Effective:* The proposed attack architecture is evaluated using five potent transformer models as they possess an inherent self-attention mechanism (includes BERT, DistilBERT, ALBERT, RoBERTa & XLNet) as well as on the popular Word-CNN & LSTM models trained on dataset. When compared to earlier baseline attack methods, the newly proposed attack method has done relatively well in terms of its imperceptibility to human observation and higher attack success rate.
  - *Simple:* In contrast to prior studies that employed techniques such as deletion, insertion, swapping, synonym substitution, misspelling, or multiple linguistic-driven processes, the proposed method creates adversarial sequences using straightforward character-level modifications.
  - *Small perturbations:* HOMOCHAR can generate adversarial inputs that retain semantic similarity for a human observer with minor visual alterations.
- Using the HOMOCHAR attack and the baseline adversarial attack techniques, the current investigation has compared and examined the performance of current sentiment classifiers in adversarial situations. The objective is to test which model stands most vulnerable and which is more robust under hostile settings. Therefore, this effort aims to address the following research question: How vulnerable is sentiment classification to adversarial attacks?

Organization: The article is organized as follows: Section 2 discusses the background and previous related works. Section 3 explains the proposed approach. The configuration of experimental settings is described in Section 4, and the results of the experiments are presented in Section 5. Section 6 provides an extensive examination of the proposed framework. Section 7 examines prospective future exploration areas and limitations of the work. The concluding section, Section 8, summarizes the main findings and derives conclusions from the presented research.

2. Related work

This section describes the sentiment classification problem and its use in various applications. Then, it discusses the background of adversarial machine learning in NLP, i.e., the art of fooling the machines with simple perturbations in text. The section also sets out some previously related cutting-edge attack methodologies in text domain.

2.1. Sentiment classification

Sentiment analysis, also referred to as analysis of opinions, is a technique utilized in the business realm to identify and categorize content based on the sentiment a block of text conveys (Derakhshan and Beigy, 2019). This content may include tweets, remarks, criticisms, and even impassioned rants containing mixed or neutral views. Monitoring client feedback, identifying specific consumers to improve service, and observing how a change in a product or service affects how customers feel are examples of common uses for sentiment analysis (Park et al., 2021). Monitoring client sentiment over time is helpful as well.

This platform has fundamentally changed how firms' function, from opinion polls to inventive marketing tactics. For instance, a lot of internet recommendation algorithms analyse user reviews and comments based on their emotion. These systems often categorize the reviews and comments into two or three groups before weighing the findings when rating movies and products (Dashtipour et al., 2021). Traditional strategies for machine learning, such as Decision Tree and Naïve Bayes and algorithms etc., have demonstrated significant success in predicting sentiment reviews (Lakshmi Devi et al., 2020). With the advent of deep learning techniques, there has been a notable advancement in the sophistication and intelligence of models, exemplified by examples such as of Word-LSTM, Word-CNN (Dashtipour et al., 2021) and transformer models. The models have garnered significant interest due to their exceptional accuracy scores in the realm of text classification. This study utilizes deep learning-based classification models that have demonstrated the highest level of effectiveness. This article demonstrates the significance of validating deep learning-based sentiment classifiers before using them in decision support systems by the use of practical assaults.

2.2. Adversarial attacks for classification in texts

In the domain of applied artificial intelligence, adversarial machine learning, which incorporates adversarial attacks, has gained prominence in recent years. In this section, an adversarial attack is defined and introduces certain of their characteristics. The distinctions between the previous work and the solution presented in this research is discussed. The characteristics of the previous state-of-the-art are presented in Table 1 to emphasize their differences and limitations.

2.2.1. Definitions

The subsequent notions constitute the foundational principles of adversarial attacks

- *Deep Neural Network (DNN):* A deep neural network (DNN or deep-learning model) may be described as a function that is non-linear $f_\theta : \mathbf{X} \rightarrow \mathbf{Y}$, where $\mathbf{X}$ is the input features/attributes and $\mathbf{Y}$ is the output predictions. The set of classes or objects can be discrete in nature. In the process of training a model, the symbol θ denotes the deep neural network parameters, which are acquired through the utilization of gradient-based back-propagation. The attainment of optimal parameters can be accomplished through the minimization of the disparity between the predicted value $f_\theta(\mathbf{X})$ of the model and the accurate label $\mathbf{Y}$. This disparity is evaluated by the loss function $\mathbf{J}$ $(f_\theta(\mathrm{X}), \mathrm{Y})$.
- *Perturbations:* Perturbations are purposefully manufactured tiny disturbances that are introduced to the original input data instances during the testing stage in order to deceive the deep-learning models.

Table 1
Comparison of baseline adversarial attack approaches with the proposed methodology. (**WB**: White-box, **BB**: Black-box; **UT**: Untargeted, **T**: Targeted; **Greedy-WIR**: Greedy word importance ranking).

Attack method	Attack specificity	Perturbation level	Attacker's knowledge	Modification	Search technique	Language constraints	Limitation
A2T (Yoo and Qi, 2021)	UT	Word-level	WB	Counter-fitted word embedding swap (or) BERT Masked Token Prediction	Greedy-WIR	Cosine similarity, Part of speech match, Word Embedding Distance	Token substitutions are sometimes out-of-context & complicated, yet are clearly detectable by humans.
Bae (Garg and Ramakrishnan, 2020a)	UT	Word-level	BB	BERT Masked Token Prediction	Greedy-WIR	USE sentence encoding cosine similarity	Because word level disturbances may be easily identified using multiple detection strategies, resilience is missing.
Check-List (Ribeiro et al., 2020)	UT	Word-level	BB	Word Swap Contract, Word Swap Extend, Word Swap Change Name Word Swap Change Number, Word Swap Change Location	Greedy Search	Repeat word Modification	The attack technique is less effective, and the number of changes made to input sequences is substantially higher.
Deepwordbug (Gao et al., 2018)	UT	Char-level	BB	{Character Deletion, Character Substitution, Character Insertion, Neighbouring Character Swap}	Greedy-WIR	Levenshtein edit distance	The alterations made to legitimate input can be easily circumvented using a spelling and grammar checker.
IGA (Wang et al., 2021)	UT	Word-level	WB	Counter-fitted word embedding swap	Genetic Algorithm	Word embedding distance, Percentage of words perturbed	The alterations made to input are sometimes readily detectable by humans
Kuleshov et al. (2018)	T, UT	Word-level	WB	Replaces with counter-fitted word embedding term	Greedy word replacement	Cosine similarity, Thought vector encoding, Language model similarity probability	The adversary must have access to the model, the feature set of inputs, and the model parameters in a white box method.
Pruthi et al. (2020)	UT	Char-level	BB	{Insertion, Character Elimination, Neighbouring Character exchange, Keyboard-Based Character Swap}	Greedy search	Minimum word length, Maximum number of words perturbed.	Using an orthography and grammar checker, it is straightforward to avoid the alterations made to valid input.
Input-reduction (Feng et al., 2018)	T, UT	Word-level	WB	Word deletion	Greedy-WIR	Minimum word length, number of words deleted	Inevitably, removing any word from a valid sequence might alter the semantic significance of the text input.
PWWS (Ren et al., 2020)	UT	Word-level	WB	WordNet-based similar word replacement	Greedy-WIR (saliency)	Sentence encoding cosine similarity	In a white box technique, an attacker must have access to the model parameters.

(*continued on next page*)

- An adversarial attack involves intentionally modifying the input data of a neural network to assess its ability to maintain its output under such conditions. The present study involves a collection of n sentences, denoted as $\boldsymbol{X} = \{x_1, x_2, \ldots, x_n\}$, along with a corresponding

Table 1 (*continued*).

Attack method	Attack specificity	Perturbation level	Attacker's knowledge	Modification	Search technique	Language constraints	Limitation
PSO (Zang et al., 2020)	UT	Word-level	WB	How-Net Word replacement	Particle Swarm Optimization	Maximum number of words perturbed, Minimum word length	This search strategy employs a time-consuming procedure to identify potential disruptions.
TextFooler (Jin et al., 2019)	UT	Word-level	BB	Counter-fitted word embedding replacements	Greedy-WIR	Part-of-speech match, Word Embedding Distance, USE sentence encoding cosine similarity	This type of attack can be readily defended by robust adversarial training that uses lexical alternatives to the words in an input text.
TextBugger (Li et al., 2019)	UT	Char-level	BB	{Neighbouring Character replacement, Elimination, Substitution, Insertion for characters}	Greedy-WIR	USE sentence encoding cosine similarity	The modifications that are made to the input sequence sometimes become noticeable.
Iyyer et al. (2018)	UT	Sentence-level	BB	Generate paraphrases that adhere to the target's specifications without sacrificing quality	–	Grammatical constraints, word part of speech checker	Paraphrasing the whole input sentence to produce hostile input may at times alter the semantic meaning.
Liang et al. (2018)	UT	Hybrid perturbation (word & char mixed)	WB, BB	Perturbation strategies: insertion, modification, and removal	Greedy search	Minimum word length, Maximum number of words perturbed	Imperceptibility constraints are lost when the modifications made to the input sequence on both word and character levels
HOMOCHAR (Proposed Approach)	UT	Char-level	BB	Word replaced with the character-level agitated word (visually similar characters known as homographs)	Beam search	Minimum cosine similarity, Part of speech consistency	–

HOMOCHAR: Character-level perturbed word (including homoglyph letters) in place of original word. It incorporates beam search to determine the best group of potentially disturbed words in this case. Keeping the key modifications under particular grammatical and semantic similarity constraints can deceive the model into making accurate predictions.

set of n labels, $\boldsymbol{Y} = \{y_1, y_2, \ldots, y_n\}$. The textual input field $\boldsymbol{X}$ is linked to the label space $\boldsymbol{Y}$ through a pre-existing model known as $\boldsymbol{f}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$. An authentic adversarial instance, denoted as $\boldsymbol{x_{adv}}$, pertaining to the expression $\boldsymbol{x} \in \boldsymbol{X}$, must satisfy the criteria outlined in Eqs. (1) and (2).

$$f(\boldsymbol{x_{adv}}) \neq f(\boldsymbol{x}) \tag{1}$$

$$Sim(\boldsymbol{x_{adv}}, \boldsymbol{x}) \geq \in, \tag{2}$$

The symbol "∈" denotes the minimum degree of similarity between the adversarial and genuine samples, while the function Sim: $\boldsymbol{X} \times \boldsymbol{X} \rightarrow \boldsymbol{(0, 1)}$ represents an analogy function. *Sim* **(.)** is a function which is frequently employed for the purpose of detecting similarities in both semantics and syntax within the realm of textual data. The intuition behind this is to consider $\boldsymbol{f}$ as a classification model which is trained with clean inputs and supposing x to be a valid input. Then it is modified from $\boldsymbol{x}$ to $\boldsymbol{x'}$, such that $\boldsymbol{x'} = \boldsymbol{x} + \boldsymbol{\delta}$, where $\boldsymbol{\delta}$ is the perturbation required for $\boldsymbol{x}$ to cross the decision boundary of true class entity, resulting in $\boldsymbol{f}(\boldsymbol{x}) \neq \boldsymbol{f}(\boldsymbol{x'})$.

2.2.2. Threat model

The notion of threat Model from the study (Yuan et al., 2019) to attack DNN is utilized. In the following section, numerous aspects of the threat model are examined which includes Attack specificity, Attacker's Knowledge and Perturbation level. The presented visual representation in Fig. 1 illustrates the categorization of adversarial attacks within the textual domain, utilizing the aforementioned disciplines as a basis for classification.

- *Attack Specificity:* Attack specificity refers to an attack in which the attacker seeks to target a fixed label after applying a perturbation or simply misclassifying the actual label, which is known to be as targeted and untargeted attack respectively. Let us primarily consider the input class as $\boldsymbol{C_i}$ and target class as $\boldsymbol{C_t}$. The input $\boldsymbol{x}$ belongs to the class $\boldsymbol{C_i}$ and after perturbation, the attacker wishes $\boldsymbol{x'}$ set belongs to the class other than $\boldsymbol{C_i}$. In a *targeted attack*, the input data $\boldsymbol{x}$ is perturbed to $\boldsymbol{x'}$ such that instead of its true class $\boldsymbol{C_i}$ it would predict a targeted class $\boldsymbol{C_t}$. Here, the concern is about hitting a specific destination. In an *untargeted attack*, the concern is to move input $\boldsymbol{x}$ away from its actual class $\boldsymbol{C_i}$, no matter which other class it strikes, as shown in Fig. 2.
- *Attacker's Knowledge:* Another criterion for categorizing adversarial attack is on basis of the knowledge of the attacker which is whether they make black-box or white-box assumptions about their adversaries. Among other degrees of knowledge, an adversary may have comprehensive knowledge of the model it is attempting to deceive, or it may have no knowledge at all. In a scenario commonly referred to as "*black-box*", the adversary's capabilities are restricted to making queries to the target classifier, without access to the intricacies of the trained models or the feature representations of inputs. Due to the lack of knowledge regarding the feature set, the attacker is limited to modifying input samples and subsequently testing and

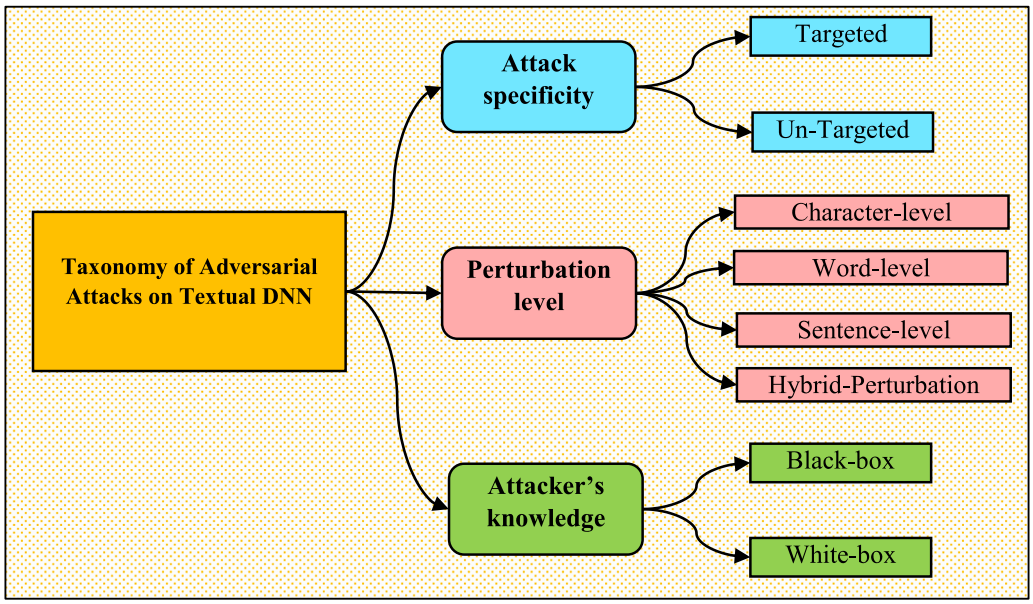


Fig. 1. Taxonomy of adversarial attacks methods in textual domain.

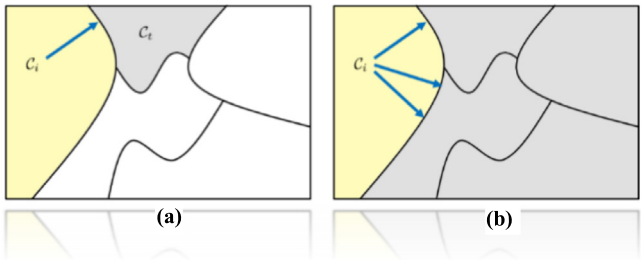


Fig. 2. (a) Targeted attack: Specific attack from C_i to C_t. (b) Untargeted attack: The attack vector can point to anywhere outside C_i.

observing the outputs. In the context of system architecture, a "*white-box*" design refers to a scenario in which the adversary has full access to the model, the input feature set, and the model parameters.

- *Perturbation level:* An adversarial attack in NLP can be executed by perturbing at different levels, including word, character, sentence, and hybrid perturbations. The process involves various techniques such as insertion, swapping, deletion, synonym replacement and mis-spelling. The aforementioned methodologies are implemented in a way that upholds the intended semantic of the sentence for human perceivers, yet induces the model to generate imprecise outcomes.

2.2.3. Textual adversarial attack framework

An adversarial attack framework seeks a perturbation of an input sequence that fulfils the assault's purpose while adhering to specific language limitations. Attacking an NLP model may therefore be presented as a combinatorial search method. The attacker is required to explore every conceivable alteration in order to identify a series of alterations that culminates in the creation of a viable adversarial instance. Each adversarial assault consists of four fundamental elements (Morris et al., 2020). Variations to the attack methodology include modifications to the input sequences, search methodology, constraint set, and attack specificity as described below.

- *Modifications*: Modification-based method for generating multiple potential permutations from a single input. Insertion of random characters, word embedding word exchange & thesaurus word replacements are examples.
- *Search technique:* A search methodology iteratively examines the model and selects favourable alterations from a range of modifications (Yoo et al., 2020). These include Greedy search, beam search, genetic algorithm, and Greedy-WIR (which involves ranking the importance of words).
- *Set of language constraints:* A set of limitations is employed to evaluate the authenticity of a disturbance with respect to the initial input (Wang et al., 2023). Examples: minimum sentence encoding cosine similarity, part-of-speech consistency, maximum word embedding distance grammar checker.
- *Specificity:* A task-specific objective function that evaluates the attack's efficacy in terms of model outputs. Examples: targeted classification, untargeted classification, non-overlapping output.

Table 1 enumerates noteworthy textual adversarial attack techniques across six disciplines, in accordance with the specifications outlined in Sections 2.2.2 and 2.2.3. It includes: *Attack specificity*, *Attacker knowledge*, *perturbation level*, *modification*, set of *language constraints*, and *searching technique* which clearly demonstrates uniqueness of each attack methodology in conjunction with the proposed approach along with the limitations in previous baselines.

Several NLP adversarial attacks utilize rule-based techniques for substituting synonyms in order to create adversarial examples. These tactics have the potential to generate token substitutions that are both contextually inappropriate and excessively intricate, yet remain readily discernible by human observers. Observing the limitations of current attack methods from the attacker's perspective, this paper presents HOMOCHAR, a potent adversarial NLP attack. This attack technique generates adversarial instances by perturbing a textual input at the character level. The ultimate goal of the attack method is to mislead a target NLP model, provided that the perturbation satisfies certain linguistic constraints. (e.g., semantic similarity constraint, grammar constraint). This investigation focusses on a black-box scenario, as black-box attack approach is considered to be a more practical method as attackers lack knowledge of the learned models or feature representations of inputs. The introduced technique is directed towards sentimental analysis services in NLP-related tasks, the method outperforms the efficacy and effectiveness of widely used baseline attack methods, i.e., by increasing attack success rates with more semantic similarity for human reader. The methodology that has been introduced will be expounded upon in the following section.

3. Proposed approach

This section provides a detailed discussion of the proposed "**HOMOCHAR**" adversarial attack framework. In the first portion of this section, the problem is outlined. The architecture of the attack is then presented, consisting of four modules that are described in greater detail. In addition, as part of the second strategy, the current study compared and examined which neural sentiment classifier is most susceptible to adversarial perturbations and which is more resistant.

3.1. Problem definition

The objective of a proficient DNN classifier **F** is to accurately predict the label $\boldsymbol{Y}_{true} \in \boldsymbol{y}$ for any given input $\mathbf{X} \in \boldsymbol{x}$, i.e., $\mathbf{F}(\mathbf{X}) = \boldsymbol{Y}_{true}$, This is achieved by maximizing the posterior probability, as demonstrated in Eq. (3).

$$\text{argmax}_{Y_i \in y} \mathrm{P}\left(\mathbf{Y}_i/\mathbf{X}\right) = \mathbf{Y}_{true} \tag{3}$$

The objective of a rational text attack is to introduce a perturbation $\Delta \mathbf{X}$ that is imperceptible to humans, but has the ability to deceive the classifier **F** when it is incorporated into the original **X**. The modified input $\mathbf{X}^* = \mathbf{X} + \Delta \mathbf{X}$ is referred to as the adversarial example in the literature. In general, an adversarial example that is successful has the ability to deceive a well-trained classifier into assigning an incorrect label that is different from the true label or a pre-specified label $\mathbf{Y}_{target}$, where $\mathbf{Y}_{target} \neq \mathbf{Y}_{true}$. Several techniques are employed to achieve the objective of rendering the generated $\mathbf{X}^*$ indiscernible, including measures such as similarity in meaning. This is done to ensure that the standard deviation of the distinction between the original and modified text, must be less than a certain threshold value, denoted by δ. The symbol δ represents a threshold that limits the number of manipulations.

$$\text{argmax}_{Y_i \in y} \mathrm{P}\left(\mathbf{Y}_i/\mathbf{X}^*\right) \neq \mathbf{Y}_{true} \tag{4}$$

$$\text{argmax}_{Y_i \in y} \mathrm{P}\left(\mathbf{Y}_i/\mathbf{X}^*\right) = \mathbf{Y}_{target} \tag{5}$$

Eqs. (4) and (5) represent the attack strategies commonly referred to as untargeted and targeted attacks, respectively. A text perturbation that is considered valid must adhere to semantic, grammatical and lexical and constraints. This paper presents a novel framework for adversarial attack, which exhibits the ability to produce adversarial text. According to the research findings, the proposed method for generating adversarial instances yields imperceptible alterations that present a formidable challenge for human observers, as they are unable to discern the perturbations.

3.2. Attack design

An effective textual adversarial approach **HOMOCHAR** is developed under black-box environment that formulates stronger adversarial examples as a combinatorial search task with the goal (untargeted attack) for deceiving neural text classifier by perturbing at character-level which adheres to specific linguistic constraints. The attack is built using four essential components, which include *transformation* (that generates a list of potential $\mathbf{X}_{adv}$(adversarial samples), *search method* (that applies transformation until a successful $\mathbf{X}_{adv}$ is found), a *set of constraints* (that filter out $\mathbf{X}_{adv}$ that does not satisfy lexical, grammatical, and semantic constraints), and a *goal function* (which assesses the effectiveness of the method such that it always misclassify the true prediction), the design of proposed methodology is shown in Error! Reference source not found.. The algorithm looks for potential changes that could lead to a successful perturbation.

This section delineated the methodology for creating adversarial examples through a framework consisting of four distinct elements: transformations, a set of constraints, a search algorithm and a goal function. The aforementioned system is intended to detect a change from **X** to **X′** that deceives a predictive NLP model. This is accomplished by satisfying specific constraints while simultaneously achieving a particular objective, such as misleading the model into producing an incorrect classification label. The search algorithm aims to identify a sequence of alterations that result in a positive perturbation outcome. The following discourse provides a comprehensive elucidation of the four fundamental sets of components utilized in the construction of the proposed methodology.

3.2.1. Transformations

From an input, a transformation generates a number of prospective perturbations. If $\boldsymbol{x} = (\mathbf{X}_1, \ldots, \mathbf{X}_i, \ldots, \mathbf{X}_n)$, then replacing $\mathbf{X}_i$ with a changed version of $\mathbf{X}_i'$ will result in a perturbed text. Depending on the granularity of $\mathbf{X}_i$, the alteration may take place at the word, character, or sentence level. Since word substitution is a common literary technique, in this research, it is decided to concentrate the investigation on swapping the important words with character-level perturbed words. In HOMOCHAR adversarial attack, the individual characters of the significant words in an input text are transformed. In transformation, normal characters are replaced with homoglyphs[1] characters (for instance, changing all English "**a**" in a neural text sample to Cyrillic "**а**"). These are chosen because, while they look visually similar to their counterparts, neural text classifiers tokenize them differently (Boucher et al., 2021). Therefore, normal characters of the most important words in an input sentence are substituted with homoglyph characters. Homoglyph characters are also named as homographs. In the past, homoglyphs have been used to substitute similar-looking characters in a trusted URL to transfer users to malicious websites. This research includes an experiment to explore if this technique can also be utilized to develop efficient black-box adversarial attacks against neural sentiment classifiers.

Homoglyph characters represent the same glyph or an identical-looking glyph. Typically, this occurs when the same written script is used in various language families. These characters are different according to the Unicode specification. A character set called Unicode[2] was created to standardize how text is represented electronically. At the current time, the character capacity of the system is 143,859 and it can accommodate diverse languages and symbol sets. Traditional Chinese characters, mathematical symbols Latin letters and emojis are just a few of the characters that can be represented by Unicode. Each character is assigned a code point, which is a numeric representation. There are other ways to encrypt these numerical code points, which are commonly identifiable by the prefix U+, but UTF-8 is the most widely used. The Unicode specification poses a significant security threat due to the diverse encoding options available for homoglyphs, which are distinct characters that exhibit identical or comparable glyphs. For instance, the look-alike digit zero 0 (U+0030) is used in place of the Latin minuscule letter O (U+006F), which will cause the computer to tokenize it differently while classifying the data. This issue is not unique to Unicode. As an illustration, within the ASCII range, the lowercase Latin character "l" frequently bears a resemblance to the uppercase Latin character "I". Certain character combinations can function as pseudo-homoglyphs, as exemplified by the "rn" and "m" pairing in most sans serif fonts.

The fundamental idea behind this transformation is to add noise by exchanging English characters for corresponding international characters, as seen in Fig. 3. Such noise may lead to classification errors that differ from the actual results (Wolff, 2020). Text appears the same to humans but yields different results for deep-learning sentiment classifiers. The findings of the research indicate that the transformed adversarial instances exhibit a high degree of imperceptibility to human visual perception, which poses a challenge for users to accurately discern the attack as being adversarial in nature. It demonstrates that HOMOCHAR attack methodology is far better compared to the standard attack techniques, which include misspellings, insertions, switching, synonym replacements, etc.

[1] IDN homograph attack - Wikipedia.

[2] https://www.unicode.org/versions/Unicode13.0.0/.

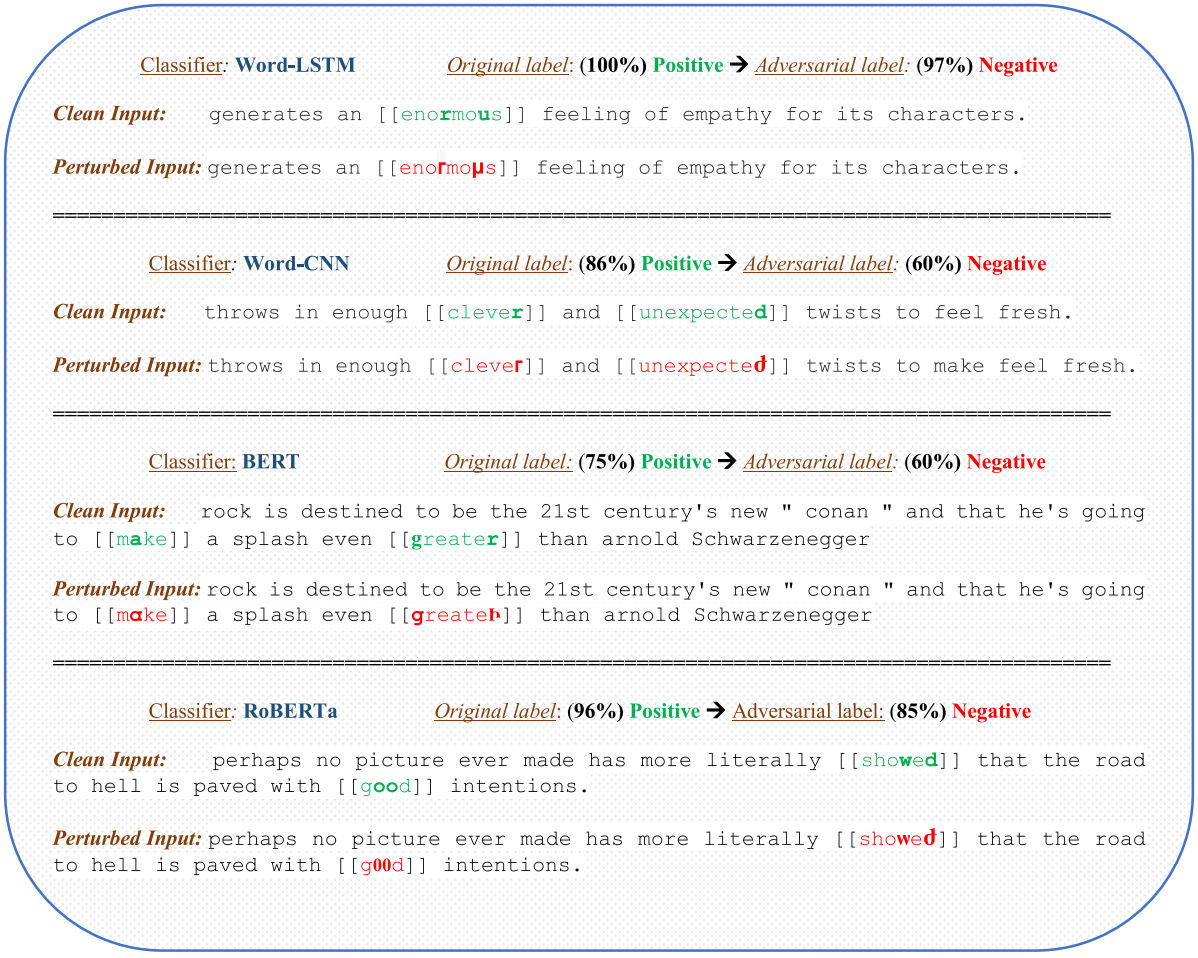


Fig. 3. Adversarial Examples generated by replacing normal English characters with visually similar homoglyphs.

3.2.2. Searching algorithm

The search algorithm attempts to identify, from the transformations, the set of most potent perturbed words in an input sequence that will result in the most efficient attack. The text $\boldsymbol{x}$ is subjected to various perturbations by substituting each word $\mathbf{X}_i$, resulting in multiple perturbed texts $\mathbf{X}'$. The beam search algorithm is utilized in order to find out the best set of perturbed sequence which uses the scoring function mentioned in Section 3.2.2.1 of this article. The words that achieved the highest scores adhering to perturbations are selected. In beam search, the top **b** most potent perturbed texts are kept (**b** is known to be the "beam width") with $\mathbf{b} = \mathbf{8}$. Subsequently, the aforementioned procedure is reiterated through perturbation of each of the highest-ranking **b** texts, resulting in the production of the subsequent group of candidates. This process requires $\mathbf{O(b} * \boldsymbol{W}^{\mathbf{2}} * \mathbf{T)}$ queries, where $\boldsymbol{W}$ denotes the quantity of words present in the input. The variable **T** represents the upper limit of available transformation choices for a specific input.

3.2.2.1. Heuristic scoring function. In the event of an untargeted attack on a classifier, the perpetrator's objective is to identify instances that cause the classifier to inaccurately predict the class (label) for $\mathbf{X}'$. The underlying premise is that the veritable classification of $\mathbf{X}'$ corresponds to that of the initial **X**. The heuristic scoring function computes the score of every element $\mathbf{X}_i$ that belongs to the set $\boldsymbol{x}$. To identify the optimal group of prospective candidates for perturbation. The candidates who receive the highest scores are selected over other candidates.

Typically, a heuristic scoring function is employed, where the score is defined as shown in Eq. (6):

$$Score\left(\mathbf{X}'\right) = 1 - F_y\left(\mathbf{X}'\right) \tag{6}$$

3.2.3. Semantic similarity

A 'fine-grained metric' is required that quantifies the extent to regulate the quality of generated adversarial texts, such that it can be contended that the produced adversarial texts will preserve semantic similarity with the original texts. The HOMOCHAR framework utilizes the Universal Sentence Encoder (USE) to evaluate the semantic similarity among textual instances (Cer et al., 2018). The USE model utilizes a process of encoding distinct input sentences into embedding vectors of 512 dimensions, thereby facilitating the computation of their cosine similarity score. Eq. (7) defines the cosine similarity between two n-dimensional vectors, denoted as $\boldsymbol{a}$ and $\boldsymbol{b}$.

$$\boldsymbol{S}(\boldsymbol{a},\boldsymbol{b}) = \boldsymbol{\delta} = \frac{\boldsymbol{a}\cdot\boldsymbol{b}}{\|\boldsymbol{a}\|\cdot\|\boldsymbol{b}\|} = \frac{\boldsymbol{\Sigma}_{i=1}^{n}\boldsymbol{a}_i \times \boldsymbol{b}_i}{\sqrt{\sum_{i=1}^{n}\left(\boldsymbol{a}_i\right)^2} \times \sqrt{\sum_{i=1}^{n}\left(\boldsymbol{b}_i\right)^2}} \tag{7}$$

The USE encoder employed in this study has been trained on a diverse range of web-based textual data with a broad scope, including but not limited to Wikipedia, web-based news articles, web-based question-and-answer pages, and online discussion forums. Thus, it possesses the ability to provide input for numerous subsequent tasks. Formally, denote USE encoder by Encoder, then the USE score between an example $\boldsymbol{X}$ and its adversarial variation $\boldsymbol{X}'$ is defined in Eq. (8) below.

$$USE_{\text{score}} = \text{Cosine}(\text{Encoder}\ (\boldsymbol{X}),\ \text{Encoder}\ (\boldsymbol{X}')) \tag{8}$$

Given that the primary objective is to effectively produce adversarial texts, it suffices to regulate the semantic similarity to a predetermined threshold ($\boldsymbol{\delta}$), and a threshold of $\boldsymbol{\delta} = \mathbf{0.7}$ is selected. The Universal Sentence Encoders (USE) is utilized to conduct a comparison between the sentence encodings of the original text denoted as $\boldsymbol{X}$ and the

Table 2
Algorithm of the proposed framework.

Algorithm 1: HOMOCHAR Adversarial Attack

Aim: Adversarial attack framework to fool neural text classifier
Input: legitimate input $\boldsymbol{X}$ and its ground truth label $\boldsymbol{Y}$, classifier $\boldsymbol{F}(.)$, threshold $\boldsymbol{\delta}$, semantic similarity $\boldsymbol{S}(.)$.
Output: Generated an adversarial sequence $\boldsymbol{X}_{adv}$

1. Initialization: $\boldsymbol{X}^* \Leftarrow \boldsymbol{X}$
2. **for** $\boldsymbol{X}_i$ in $\boldsymbol{x}$ **do**
3. Compute score ($\boldsymbol{X}_i^*$) according to **Eq. (5)**
4. **end for**
5. $\boldsymbol{W}_{Ordered} \Leftarrow \boldsymbol{Sort}(\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3 \ldots\ldots\ldots\ldots \boldsymbol{X}_n)$ in descending order
6. Remove the stop words in $\boldsymbol{W}_{Ordered}$
7. **for** $\boldsymbol{X}_i$ in $\boldsymbol{W}_{Ordered}$ **do**
8. $\boldsymbol{X}^* \Leftarrow \boldsymbol{replacing\ X\ with\ (words\ containing\ homoglyphs)}$
9. **if** $\boldsymbol{S}(\boldsymbol{X}, \boldsymbol{X}^*) <= \boldsymbol{\delta}$ **then**
10. return None:
11. **else if** $\boldsymbol{F}(\boldsymbol{X}^*) \neq \boldsymbol{Y}_{true}$, **then**
12. Solution found. return $\boldsymbol{X}^*$.
13. **end if**
14. **end for**
15. **return None**

Search method
Transformation
Constraint
Goal Function

perturbed text denoted as $\boldsymbol{X}'$. In the event that the cosine similarity of two encodings decreases to a specific threshold, the value of $\boldsymbol{X}'$ is disregarded. The utilization of large encoders such as Universal Sentence Encoder (USE) poses a challenge due to the considerable GPU memory consumption, which can reach up to 9 GB in the case of USE (Yoo and Qi, 2021). Also, Language Tool (Naber et al., 2003) is used to induce the minimum number of grammatical errors along with Part-of-speech consistency (The substitute word should share the same part of speech as the original one.). Support taggers provided by flair, SpaCy, and NLTK attempt to preserve semantics between $\boldsymbol{X}$ and $\boldsymbol{X}'$.

3.2.4. Goal function

The efficacy of an attack is evaluated in relation to model outputs through the utilization of a goal function. It probes the search method along with transformations and a particular set of constraints until it leads to the misclassification of the actual output. The generated Adversarial Examples resulting from the proposed algorithm are depicted in Fig. 3. Table 2 illustrates the algorithm utilized in the proposed framework.

Assuming the given input document $\boldsymbol{x} = (\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n)$, where each $\boldsymbol{X}_i$ denotes input sequence located at the ith position. Initially, the spaCy library is employed to segment each document into distinct sentences. The process involves eliminating sentences that have predicted labels that differ from the original document label, specifically by filtering out $\boldsymbol{F}(\boldsymbol{X}_i) \neq \boldsymbol{Y}_{true}$. To accomplish this, the first step is to identify the most significant words that have the maximum influence on the original prediction outcomes, using a heuristic score as outlined in Eq. (5). These words are then subject to slight modifications while ensuring that their semantic similarity is maintained. The heuristic scoring function possesses three key characteristics. Firstly, it has the ability to accurately reflect the significance of words in relation to the prediction. Second, it can compute word scores without any prior understanding of the classification model's framework and settings. Lastly, it is a highly efficient method of calculation. In the development of adversarial instances, a preference is given to making small modifications to the original words. This is due to the requirement that the resulting adversarial sentence must maintain visual and semantic similarity to the original sentence, in order to facilitate comprehension by human observers.

The design decisions are made so as to generate adversarial examples with higher quality and less disturbance. In the transformation function, the process involves substituting standard English characters with homoglyph characters for the dominant terms within a given input sentence. The beam search algorithm is utilized to determine the optimal set of perturbed candidates for a successful attack. Furthermore, the semantic similarity between $\boldsymbol{X}_{adv}$ & $\boldsymbol{X}$ is measured using the Universal Sentence Encoder (Cer et al., 2018). In addition, Language Tool (Naber et al., 2003) is utilized to generate minimal grammatical errors while maintaining Part-of-Speech consistency. All of these design decisions result in constructing a robust adversarial attack method relative to the baselines which in turn results in a potent untargeted attack that misclassifies the actual output. The proposed technique involves perturbing text through the substitution of normal English letters with homoglyphs at the character level. An observation of significance is that words possess a symbolic quality, and language models conventionally depend on a lexicon to depict a finite range of possible words. The magnitude of the standard vocabulary is considerably lesser than the potential permutations of characters at a commensurate extent (e.g., approximately 26^n for the English language, wherein n denotes the word's length). This suggests that purposeful manipulation of significant terms can lead to their conversion into "unknown" words, which are not listed in the vocabulary. In deep learning modelling, any unfamiliar words will be assigned to the "unknown" embedding vector. The outcomes of this investigation offer persuasive proof that the adoption of a simple methodology can effectively prompt text categorization models to display flawed conduct.

3.3. Comparison of sentiment classifiers

To further understand the limitations of sentiment detection methods, the experiment is divided into two sections. Using the dataset, state-of-the-art deep learning-based language models are initially trained. Then the baseline textual attacks and HOMOCHAR adversarial attack is used to manipulate the trained models to categorize positive sentiment as negative and vice versa. The general framework for performing an attack on the sentiment classifier is shown in Fig. 4. This method aims to determine which model is more vulnerable to adversarial perturbations. According to the authors, this is the first piece of work to be presented in the literature that compares sentiment classifiers in order to identify their vulnerability to adversarial cases.

4. Experimental approach

A detailed description of the datasets, victim models, attack techniques, evaluation metric, and experimental settings were all explained in this part. After that, in the following section, the findings will be assessed and likely causes of the observed performance is identified.

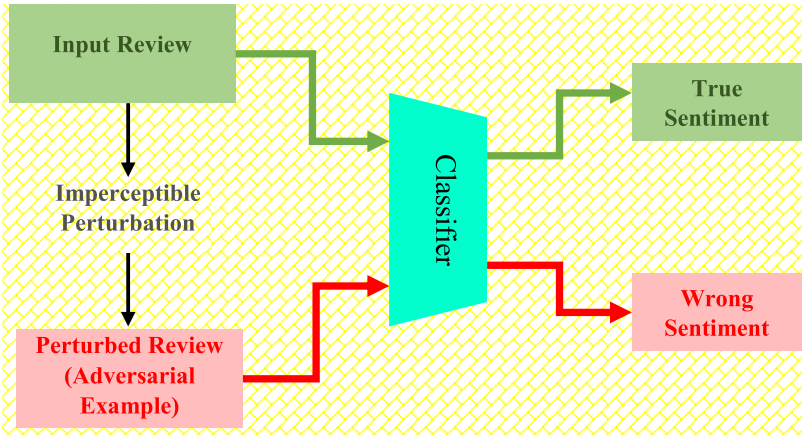


Fig. 4. Framework for conducting adversarial attack on sentiment classifier by generating adversarial test samples (adversarial examples) using various attack algorithms.

Table 3
Overview of the datasets.

Task	Dataset	Classes	Train	Test	Avg Len
Sentiment classification	**MR**	2	8.5K	2K	32
Sentiment classification	**IMDB**	2	25K	25K	215–216

4.1. Dataset description

This study investigates adversarial text samples on two publicly available benchmark datasets that are extensively used for sentiment analysis tasks. On the test set, the final adversarial examples are generated and evaluated. The Table 3 presents a summary of the datasets.

Rotten Tomatoes Movie Reviews[3] **(MR)** (Pang and Lee, 2005): The movie reviews in this dataset were gathered by Pang and Lee (2005). It has 5331 negative & 5331 positive processed sentences/snippets with an average length of 32 words. The dataset is split into three sections for the experiment, using 80% and 20% for training and testing respectively. The models were trained to perform binary classification on movie reviews, with the aim of categorizing them as either having a positive or negative sentiment.

Internet Movie Database[4] **(IMDB)** (Maas et al., 2011)**:** The dataset of movie reviews from IMDB comprises 50,000 reviews that exhibit a high degree of polarity, with 25,000 reviews allocated for training and 25,000 reviews for testing. The dataset contains an average sample length of between 215 and 216 words. The models underwent training to execute binary classification of movie reviews, with the objective of categorizing them into either a positive or negative sentiment.

4.2. Victim models

Experiments are performed on the following models to demonstrate the efficacy of the proposed framework. The descriptions of the models and their hyperparameters such as number of epochs, batch size, learning rate, sequence length etc., are provided below.

Word-LSTM: In sequence modelling, long-short term memory (LSTM Hochreiter and Schmidhuber, 1997) is frequently utilized. A 150 hidden state LSTM with bidirectional operation was designed. The input is initially converted to 200-dimensional GLoVE embeddings before being sent to the LSTM. then the final logistic regression is utilized to predict the sentiment, averaging the LSTM outputs at each timestep to produce a feature vector with a dropout set to 0.3 and achieving an *0.8070* & *0.8830* testing accuracy on MR and IMDB dataset respectively.

Word-CNN: Convolutional neural networks constitute potential strategy for text classification tasks. For the investigation, Kim's architecture of convolutional neural network model (Kim, 2014) is choosen. Word-CNN with 100 filters and 3 window sizes (3, 4, and 5) is utilized. Model dropout is set at 0.3, and a base of the 200-dimensional GLoVE embeddings is used, followed by a fully connected, max-pooling over time layer for classification. The model is achieving 0.*7940* & *0.8630* accuracy on the test set for MR and IMDB dataset respectively.

BERT: BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) employs a Masked Language Model (MLM) & Next Sentence Prediction (NSP); it uses a stack of self-attention and fully connected layers to encode a sentence. The BookCoorpus and English Wikipedia datasets served as the first training grounds for the BERT. In this study, for the sentiment classification task, the "bert-base-uncased" model underwent five iterations of training, utilizing a batch size of 16, a learning rate of 2e–05, and a maximum sequence length of 128. The aim of this optimization was to enhance its performance in sequence classification on the given dataset. Given that this was a classification problem, a cross-entropy loss function was used to train the model. The model's highest score on this job, as determined by the evaluation set accuracy, was *0.875234* & *0.89088* for MR & IMDB respectively, which was discovered after 4 epochs for both the datasets.

DistilBERT: DistilBERT (Sanh et al., 2019) is a transformer model that was derived from the BERT base. It is characterized by its compactness, speed, efficiency, and lightweight nature. In comparison to bert-base-uncased, the aforementioned model exhibits a 60% increase in speed and a 40% reduction in parameters. Despite these optimizations, it sustains a performance level of over 95% in relation to BERT, as assessed by the GLUE language comprehension benchmark. The model known as "distilbert-base-uncased" underwent training for sequence classification, lasting three epochs. The training process employed a batch size of 128, a learning rate of 2e–05, and a maximum sequence length of 16. The model was trained using a cross-entropy loss function because this task involved classification. The evaluation set accuracy, determined after two epochs, revealed that the model achieved a maximum score of *0.839587* and *0.88* on the MR and IMDB datasets, respectively.

ALBERT: BERT base has 110 million parameters, which makes it computationally expensive, a light version with fewer parameters was needed. The ALBERT (Lan et al., 2020) model comprises 128 embedding layers, 768 hidden layers, and 12 million parameters. The lighter model lowered the training and inference times as expected. Cross-layer parameter sharing and factorized embedding layer parameterization are the 2 strategies used to achieve a smaller set of parameters. The

[3] https://huggingface.co/datasets/rotten_tomatoes.
[4] https://huggingface.co/datasets/imdb.

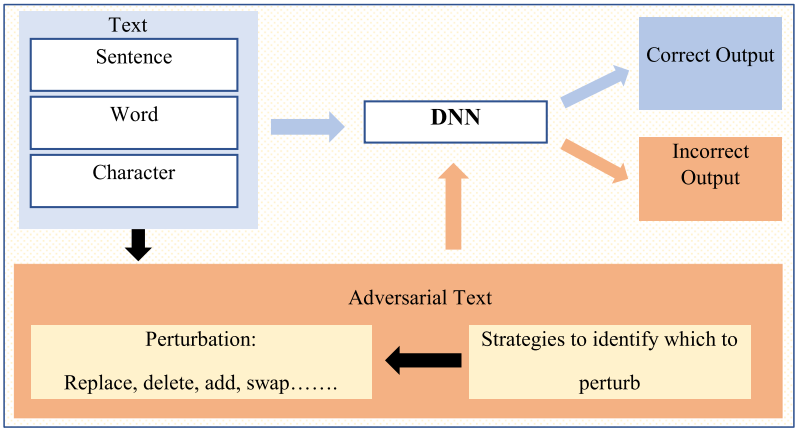


Fig. 5. Ultimate objective of any attack is to misclassify the output using different strategies.

"albert-base-v2" model was improved for sequence classification. Running it for 5 epochs with a 32-batch size, a 2e–05 learning rate, and a 128-bit maximum sequence length. A cross-entropy loss function was used to train the model. The model's highest score on this job, as determined by the evaluation set accuracy, was *0.880863* on MR which was discovered after one epoch & *0.89236* for IMDB following three epochs.

RoBERTa: Robustly Optimized BERT pre-training Approach is called RoBERTa (Liu et al., 2019). This is, in many ways, an improved form of the BERT model as it incorporates the idea of dynamic masking, which strengthens the model. In addition, RoBERTa had also been trained on datasets which include CC-News (Common Crawl-News), Open WebText, and others. These datasets have a combined size of about 160 GB. RoBERTa used a batch size of 8000 with 300,000 steps to increase the model's speed and efficiency. BERT, in contrast, use a 256-batch size with 1 million steps. The "Roberta-base" model has been fine-tuned for sequence classification. Running it with a maximum sequence length of 128 and a batch size of 32 for 10 epochs with a 5e–05 learning rate. Given that this was a classification problem, a cross-entropy loss function was used to train the model. The model's highest score on this job, as determined by the evaluation set accuracy, was *0.903377*, which was discovered after 9 iterations for MR dataset & *0.91436* was attained within 2 epochs for IMDB.

XLNet: Transformer-XL, the most advanced autoregressive model, is incorporated into XLNet's pretraining. Empirically, on 20 tasks, XLNet (Yang et al., 2019) outperforms BERT in similar experimental conditions. The design of BERT is comparable to that of XLNet. The way pre-training is handled where it differs the most, though. In contrast to BERT, which is based on autoencoding (AE), XLNet is an autoregressive model (AR). The MLM challenge makes this disparity clear by requiring the model to predict language tokens that have been randomly disguised. The "Xlnet-base-cased" model was improved for sequence classification. Using a cross-entropy loss function, it was run for 5 epochs with a batch size of 16, a learning rate of 2e–05, and a maximum sequence length of 128. The evaluation set accuracy, which was discovered after two epochs, indicated that the model's best performance was *0.907129* on MR and *0.95352* for IMDB after running it for 2 epochs.

The accuracy score is the metric employed to assess binary sentiment classification models. The Table 4 displays the range and description of the formula of accuracy. The accuracy of the target models on the standard test set is presented in Table 5.

4.3. Baseline attack methods

The attack approaches were applied to the dataset to formulate adversarial examples. Such adversarial samples are then used to manipulate the seven introduced models to classify the positive sentiment of a review as negative which results in obfuscating sentiment detection. A brief explanation of all adversarial attack approaches used in conjunction with proposed method is provided in Table 6. These attack strategies are developed by perturbing input text sequence on different levels such as word-level & character-level as shown in Fig. 5.

Table 4
Classification metric for evaluating sentiment classifiers.

Metric	Formulae	Range
Accuracy (ACC)	$\frac{TP+TN}{TP+TN+FP+FN}$	[0,1]

Table 5
Testing accuracy of the targeted models.

	Word-CNN	Word-LSTM	BERT	DistilBERT	ALBERT	RoBERTa	XLNet
MR	79.4%	80.70%	87.5%	83.9%	88.0%	90.3%	90.7%
IMDB	86.3%	88.3%	89.0%	88.0%	89.2%	91.4%	95.3%

4.4. Attack evaluation metric

The ultimate goal of attack algorithms is to alter the input in a way that leads to the model making inaccurate predictions. For accessing the effectiveness of the attack models, 500 correctly classified cases are randomly chosen from the test set, so that the accuracy of the classifiers does not affect the evaluation. The attack algorithms are then run on these source texts to produce adversarial instances. The deep learning-based sentiment classifiers are then given the adversarial instances to produce the final prediction. The percentage of incorrect predictions made by these classifiers is used to define the attack algorithm's success rate. A higher success rate indicates that the attack algorithm can produce stronger adversaries that can make these sentiment classifiers act inappropriately. The Attack Success Rate (ASR) (*ratio of successful attack samples to the sum of successful and failed samples* $\frac{successfulsamples}{successful+failed\ samples}$) metric is utilized to evaluate the efficacy of individual attack algorithms in compromising a victim model. The ASR indicates the degree to which an adversary can deceive a victim model. Formally, an attack is successful if the classifier $\boldsymbol{F}$ can accurately classify the original legitimate input $\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{Y}_{true}$, but makes an incorrect prediction for the attacked input $\boldsymbol{F}(\boldsymbol{X}+\boldsymbol{\Delta X}) = \boldsymbol{Y}^*$. Consequently, the ASR is described in Eq. (9).

$$\frac{\boldsymbol{F}(\boldsymbol{X}+\boldsymbol{\Delta X}) = \boldsymbol{Y}^*}{(\boldsymbol{F}(\boldsymbol{X}+\boldsymbol{\Delta X}) = \boldsymbol{Y}^*) + (\boldsymbol{F}(\boldsymbol{X}+\boldsymbol{\Delta X}) = \boldsymbol{Y}_{true})} \tag{9}$$

where $\boldsymbol{Y}^*$ is any label other than $\boldsymbol{Y}_{true}$ (an untargeted attack). The $\boldsymbol{\Delta X}$ indicates modifications to the legitimate text sample. A successful attack in this context means that the adversarial sample can incorrectly

Table 6
Adversarial attack algorithms in NLP.

Attacks	Perturbation	Description
TextFooler (Jin et al., 2019)	Word-level	Word swapping is used in this attacking strategy with the victims' 50 nearest embedding neighbours. optimized on BERT.
TextBugger (Li et al., 2019)	Char-level	This attack strategy's potency has been increased for use in realistic circumstances. They use character switching, space insertions, and character deletions. In context-aware word vector space, they also swap out words with their top nearest neighbours and characters with letters that seem similar (for example, o with 0).
PWWS (Ren et al., 2020)	Word-level	These attacks aim to retain lexical accuracy, grammatical correctness, and semantic closeness by leveraging synonym swap. A combination of a word's saliency score and its maximum word-swap efficacy determines its priority.
PSO (Zang et al., 2020)	Word-level	A sememe-based word replacement technique combined with particle swarm optimization for word-level attacks.
Pruthi (Pruthi et al., 2020)	Char-level	Simulates typical typos, focusing on the QWERTY keyboard. This approach uses character switching, deletion, and insertion.
Kuleshov (Kuleshov et al., 2018)	Word-level	Replaces the key words in an input sequence from counter-fitted word embedding space under a set of essential constraints.
IGA (Wang et al., 2021)	Word-level	This attack approach ranks the most crucial words in an input sequence using a scoring function and then replaces them with counter-fitted word embeddings. Along with grammatical and natural checks, it leverages word embedding distance and sentence encoding cosine similarity to maintain the validity of the perturbed sample.
Liang (Jia et al., 2019)	Word-level	Swapping words with the synonyms in the nearest word embedding space by utilizing a genetic algorithm, the aforementioned task can be accomplished under a set of potential constraints for a valid adversarial sample.
DWB (Gao et al., 2018)	Char-level	Produces minor text alterations in a black-box environment. With greedy replace-1 scoring, it employs a variety of character-swapping techniques, including swapping, substituting, deleting, and insertion.
BAE (Garg and Ramakrishnan, 2020b)	Char-level	This attack methodology uses a language model transformation with a BERT mask. To better fit the entire context, it replaces tokens using the language model.
A2T (Yoo and Qi, 2021)	Word-level	This attack approach uses gradient-based synonym word swap under white-box adversarial settings. It uses sentence encoding cosine similarity for retaining semantic similarity along with grammatical checks.
HOMOCHAR (Proposed approach)	Char-level	Word replaced with the character-level agitated word (containing homoglyph characters). In this study, beam search was employed to identify the optimal group of potentially perturbed words. Keeping the key modifications under particular grammatical and semantic similarity constraints can deceive the model into making accurate predictions.

Table 7
The study reports on the outcomes of an automated evaluation of an attack system on datasets for text classification. The evaluation includes metrics such as the accuracy of the original model's predictions prior to the attack, referred to as "**OA**" or "**Original Accuracy**", as well as the accuracy of the model following the adversarial attack, referred to as "**AAA**" or "**After-Attack Accuracy**". Additionally, the study reports on the percentage of perturbed words in relation to the original sentence length, referred to as "**PR**" or "**Perturbation Rate**".

	Word-CNN		Word-LSTM		BERT		DistilBERT		ALBERT		RoBERTa		XLNet	
	MR	IMDB	MR	IMDB	MR	IMDB	MR	IMDB	MR	IMDB	MR	IMDB	MR	IMDB
OA	79.4%	86.3%	80.70%	88.30%	87.50%	89.00%	83.9%	88.0%	88.0%	89.2%	90.3%	91.4%	90.7%	95.3%
AAA	04.1%	03.9%	01.24%	02.3%	08.4%	06.6%	0.4%	2.7%	0.2%	1.3%	3.4%	5.7%	5.4%	4.5%
PR	15.4%	11.3%	16.4%	13.2%	12.4%	11.7%	15.6%	10.3%	15.8%	12.3%	12.5%	11.8%	16.4%	09.8.%

predict with high accuracy score. In the case of a failed attack, the adversarial sample is incapable of misclassifying the actual prediction. In addition, there are statements that were omitted from the calculation. The statements that the model initially incorrectly classified during its training. The investigation focuses on the success rates of attacks and their efficacy in misclassifying outputs.

5. Results & discussion

To understand the vulnerability of a sentiment classifier. The first step of the analysis entails the utilization of the provided dataset to train advanced deep-learning models. *Section* 4.2 presents the hyperparameters of the models for both datasets with their corresponding descriptions along with their testing accuracy. The trained models are subjected to manipulation through the utilization of the HOMOCHAR adversarial attack technique. Following perturbation of the test samples by the proposed algorithm, Table 7 depicts the reduction in accuracy scores.

A total of 500 samples that were accurately classified were extracted from the test set. Subsequently, the adversarial examples are generated by employing various attack algorithms. The present study involved subjecting a set of adversarial cases to seven cutting edge sentiment classifiers. The performance of various adversarial attacks was then compared with the proposed attack in this study, using ASR as the metric. This metric can show how effective the attack strategy is. A higher ASR value indicates that a particular attack type is more effective at deceiving the model. Tables 8 and 9 present a summary of the primary outcomes of the HOMOCHAR attack method on the MR and IMDB datasets, alongside a performance comparison with previous attack techniques. Table 5 illustrates that the models under consideration exhibit commendable performance in non-adversarial scenarios. Nevertheless, the texts that are adversarially generated by HOMOCHAR exhibit a considerable attack success rate across all models. Furthermore, it has been observed that non-transformer-based models exhibit a higher vulnerability to adversarial texts in comparison to transformer-based sentiment classifiers.

The HOMOCHAR method exhibits a notable ability to perturb a limited number of words, thereby achieving a considerably high rate of attack success. This approach outperforms baseline algorithms across all models. The perturbation of only a limited number of words in samples resulted in a success rate of 97.82% on the IMDB dataset and 97.78% on the MR dataset against the Word-CNN model. For the Word-LSTM model, the proposed method achieved an ASR value of 98.96% on the IMDB and 99.64% on MR dataset with a perturbance

Table 8
Attack results on models trained on **MR** dataset. (**ASR** = Attack Success Rate & **APR** = Average Perturbed rate).

Attacks	BERT		DistilBERT		RoBERTa		ALBERT		WordCNN		WordLSTM		XLNet	
	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%
A2T (Yoo and Qi, 2021)	29.29	11.67	33.71	14.41	30.43	12.94	54.50	14.47	46.23	13.00	54.50	11.33	31.95	14.26
BAE (Garg and Ramakrishnan, 2020b)	54.55	18.79	56.18	16.07	63.04	14.26	73.54	14.19	62.06	15.15	73.54	12.78	56.70	16.61
DWB (Gao et al., 2018)	91.92	22.63	98.88	18.43	97.83	14.97	95.21	19.98	97.49	19.90	99.21	17.37	95.88	22.39
Liang (Jia et al., 2019)	56.56	17.96	61.80	17.92	62.92	18.77	75.57	17.97	77.22	16.77	78.57	14.59	59.79	19.86
IGA (Wang et al., 2021)	90.91	14.36	93.26	16.96	92.13	16.55	96.56	17.32	96.98	15.13	96.56	12.68	95.88	17.15
Kuleshov (Kuleshov et al., 2018)	90.91	13.94	**100**	13.05	98.75	13.22	98.37	12.71	97.40	12.03	98.37	11.31	95.88	14.13
Pruthi (Pruthi et al., 2020)	52.53	08.80	42.70	09.28	70.65	08.27	46.83	07.56	34.42	08.20	46.83	07.80	65.98	08.55
PSO (Zang et al., 2020)	93.94	19.99	92.13	17.80	95.65	14.09	92.41	14.94	96.40	12.03	98.41	13.04	93.81	14.73
PWWS (Ren et al., 2020)	88.89	15.12	91.01	13.76	92.39	12.37	96.30	13.19	94.97	13.00	96.30	11.74	88.66	12.59
Textbugger (Li et al., 2019)	61.62	16.86	79.78	17.92	80.43	12.80	81.08	19.61	85.07	10.55	81.08	14.12	68.04	18.29
TextFooler (Jin et al., 2019)	96.97	20.48	97.75	15.09	**98.91**	14.21	99.47	15.83	97.75	13.75	99.47	11.66	95.88	16.64
HOMOCHAR	**98.98**	15.40	**100**	14.32	**98.91**	12.67	**99.64**	16.60	**97.78**	18.45	**99.64**	17.11	**96.91**	16.23

Table 9
Attack results on models trained on **IMDB** dataset. (**ASR** = Attack Success Rate & **APR** = Average Perturbed rate).

Attacks	BERT		DistilBERT		RoBERTa		ALBERT		WordCNN		WordLSTM		XLNet	
	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%	ASR%	APR%
A2T (Yoo and Qi, 2021)	25.20	10.44	38.52	15.41	38.88	11.38	50.34	14.02	42.34	16.08	59.96	12.45	38.67	13.09
BAE (Garg and Ramakrishnan, 2020b)	48.48	16.08	58.82	18.03	70.03	16.06	77.08	13.76	70.31	13.04	70.22	14.42	49.69	15.44
DWB (Gao et al., 2018)	95.62	18.43	97.72	16.66	95.55	16.22	94.72	18.02	94.88	15.28	98.46	16.02	92.12	19.89
Liang (Jia et al., 2019)	50.04	14.26	64.44	15.78	58.58	14.32	77.21	13.88	80.81	09.44	75.78	13.00	62.86	14.32
IGA (Wang et al., 2021)	89.87	11.28	90.12	12.44	90.21	13.84	96.02	13.04	92.55	12.78	97.87	10.34	89.85	14.00
Kuleshov (Kuleshov et al., 2018)	88.62	13.42	**94.92**	11.28	94.66	11.58	97.72	12.78	97.46	13.68	97.02	11.67	96.44	13.54
Pruthi (Pruthi et al., 2020)	55.43	10.02	48.80	09.12	68.72	10.44	49.49	11.42	38.96	10.76	48.98	07.02	70.31	11.04
PSO (Zang et al., 2020)	94.76	21.38	90.10	16.66	91.92	19.00	92.68	14.48	92.12	19.49	95.84	17.78	87.82	16.68
PWWS (Ren et al., 2020)	85.52	14.44	95.62	11.64	97.77	12.28	92.21	14.43	96.64	13.42	98.20	14.00	88.42	11.24
Textbugger (Li et al., 2019)	71.84	15.88	97.92	19.42	89.92	18.22	85.07	18.07	88.87	16.67	84.33	16.69	71.12	18.68
TextFooler (Jin et al., 2019)	94.78	19.46	94.94	13.32	94.48	17.68	98.71	12.64	96.66	18.28	97.92	13.44	92.56	13.07
HOMOCHAR	**95.91**	11.28	**97.94**	09.28	**98.29**	11.44	**98.77**	11.02	**97.82**	09.33	**98.96**	13.08	**97.47**	08.88

rate of less than 18% on both datasets. In comparison, all baselines failed to surpass this success rate. The average length of the IMDB dataset is between 215–216 words. In order to conduct successful attacks, HOMOCHAR perturbed only a fraction of approximately less than 14% of the words in a single sample. The MR dataset exhibits an average length of 32 words, and in order to carry out successful attacks, HOMOCHAR perturbed only approximately 3 words for a single sample. It has been observed that HOMOCHAR has the ability to deceive transformer models, provided that the perturbation ratio remains below 20%. The ALBERT model demonstrates the highest ASR of 99.64% & 98.77% on MR and IMDB datasets respectively compared to other attack methods. The attack methodology employed in this study results in a DistilBERT model prediction accuracy of 0%, achieved through a perturbation rate of merely 14.32% on MR dataset, For IMDB dataset HOMOCHAR attains 97.94% on DistilBERT model. Despite the reputation as the top-performing model for various natural language processing tasks, BERT – a complex model with 110 million parameters – is still susceptible to HOMOCHAR adversarial attack. The findings indicate that with a perturbance rate of less than 16%, the proposed approach able to achieve an ASR of 98.98% & 95.91% on MR and IMDB respectively. Also, for RoBERTa and XLNet, HOMOCHAR outperforms prior cutting-edge attack techniques on both datasets. This indicates that the proposed attack system is capable of manipulating classifiers to generate faulty predictions.

For the purpose of evaluating the proposed attack model, the code is made available on GitHub[5] repository.

The additional goal of the research is to compare the susceptibility of different sentiment models to different types of adversarial perturbations. The objective is to determine the comparative vulnerability and resilience of various models to adversarial perturbations. On each targeted model, the attack's success rate of each attack is evaluated to determine which model is the most and least vulnerable. The calculation of the mean attack success rate for each model is determined through the utilization of Eq. (10), as illustrated below.

$$S_r = \frac{\sum_{i=1}^{a} \frac{S_i}{S_i + F_i}}{a} \tag{10}$$

S_r = *Attack Success rate*; a = *attack*; S_i = *successful attack*; F_i = *Failed attack* (The Attack Success Rate is S_r, no. of successful attacks is S_i, the no. of unsuccessful attacks is F_i, and the no. of attack recipes is a. The statements the model initially incorrectly anticipated during its training are skipped statements. They were not included in the calculation)

As illustrated in Fig. 6, Upon evaluation, it has been determined that the Word-LSTM model exhibits the highest susceptibility to adversarial attacks. Among transformer models, model BERT is the least & ALBERT is the most vulnerable; the observations conclude that lighter models are more susceptible to these attacks as the ALBERT model comprises 128 embedding layers, 768 hidden layers, and 12 million parameters but on the other hand BERT base has 110 million parameters, which is a heavy model with a very high computational complexity which makes it least vulnerable as compared to all models. The findings may be of significance to individuals who regularly employ established state-of-the-art models for their sentiment classification tasks. The reader shall be capable of ascertaining the most suitable model that aligns with their specific concern. Furthermore, this serves as a driving force for researchers to construct models that possess adversarial robust generalizations as opposed to conventional generalizations.

In order to evaluate the effectiveness and efficacy of various attack types in deceiving the model, based on their respective average perturbance rates. The mean success rate and perturbation rate for each attack type across all models on the MR & IMDB dataset are presented in Fig. 7.

6. Further analysis

An additional study is conducted to evaluate the effectiveness of the HOMOCHAR attack approach in various scenarios, including its

[5] https://github.com/Ashish250996/HOMOCHAR-adversarial-attack.

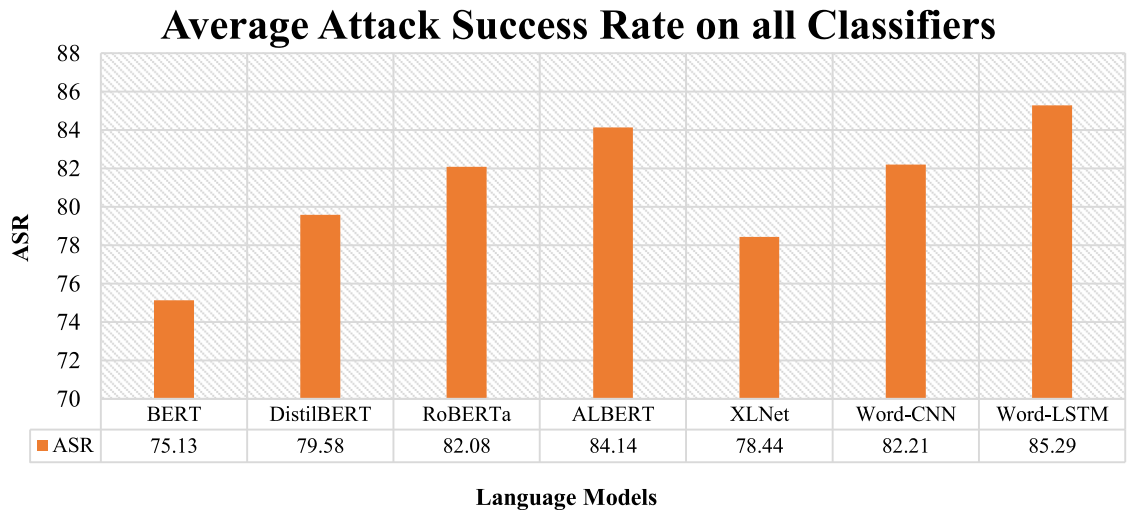


Fig. 6. Mean success rate of sentiment classifiers on all adversarial perturbations.

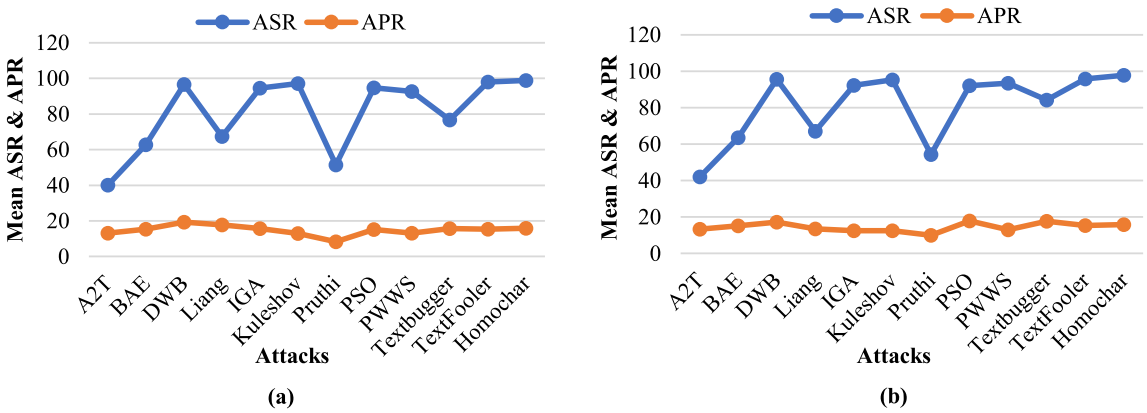


Fig. 7. Mean ASR & APR score of each attack type on all models trained on (a) MR & (b) IMDB dataset.

overall execution time in producing an adversarial example. What is the requisite memory capacity for the generation of adversarial sequences? Additionally, the effectiveness and efficiency of the proposed methodology in enabling the scalability of sample size are assessed. Furthermore, its responsiveness to alterations in the semantic similarity score is noteworthy. The evaluation of scalability and sensitivity involved an analysis of the three models that were introduced in the study, namely Word-CNN, Word-LSTM, and BERT. Furthermore, the property pertaining to the transferability of adversarial examples generated through the proposed algorithm is also evaluated. The evaluation of its efficacy in the presence of random word perturbation is also conducted. The subsequent section presents a comprehensive evaluation of the HOMOCHAR methodology, encompassing various parameters.

A. Runtime Considerations: An investigation was carried out to assess the computational time consequences of the proposed framework. From the perspective of an attacker, the primary objective is to deceive a model through the execution of the proposed attack. The potential results of the average runtime to generate a single adversarial sequence using HOMOCHAR framework for each particular model is presented Fig. 8. Empirical evidence suggests that the process of producing adversarial samples for the IMDB dataset is time-consuming. The average length of the input review is between 215-216 and 32 words for the IMDB and MR datasets respectively. Specifically, there exists a positive correlation between the time required to generate a single adversarial text and the average length of the input. As the input's size increases, the duration for producing a single adversarial text experiences a slight rise due to the increased time required to identify significant terms for perturbations.

B. Sensitivity: The norm constraint on image perturbations ($\|X - X'\|_\infty < \varepsilon$) is a crucial determinant of an attack's effectiveness in computer vision. Elevated values of the variable ε result in an increased probability of misclassification for X'. In the field of natural language processing, it is common to obtain invariance without much effort. For instance, when using a model at the word-level, the majority of perturbations generated by a character-level adversary result in an "unknown" token at the model's input. The cosine similarity function is employed in attack class for the purpose of limiting the word error rate (WER). A decrease in the cosine similarity score (δ) (*discussed in Section* 3.2.3) corresponds to an increase in the word error rate (WER), indicating that the model becomes more susceptible to perturbations. However, it can inevitably invalidate the constraint of human imperceptibility. The characteristic of a model that pertains to its responsiveness is commonly referred to as sensitivity. Therefore, in order to minimize the quantity of disturbances. In HOMOCHAR, the δ is assigned a value of **0.7** which will maintain WER such that it is sensitive to perturbations while preserving the semantic coherence of the sentence. To assess the sensitivity of a model, one must observe the fluctuations in the ASR scores in relation to variations in δ, which is defined within the range of [0.1,1) as shown in Figs. 9(b) & 10(b).

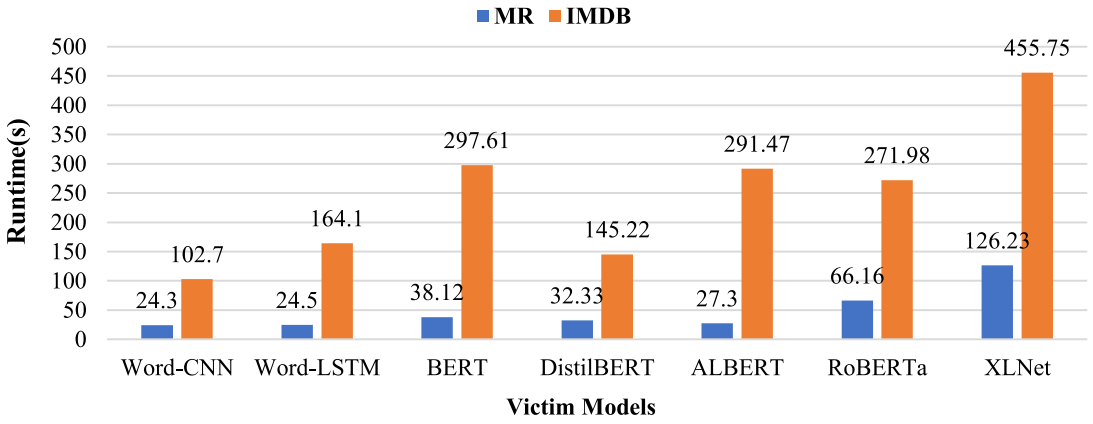


Fig. 8. Average time required for generating an adversarial sample on each model.

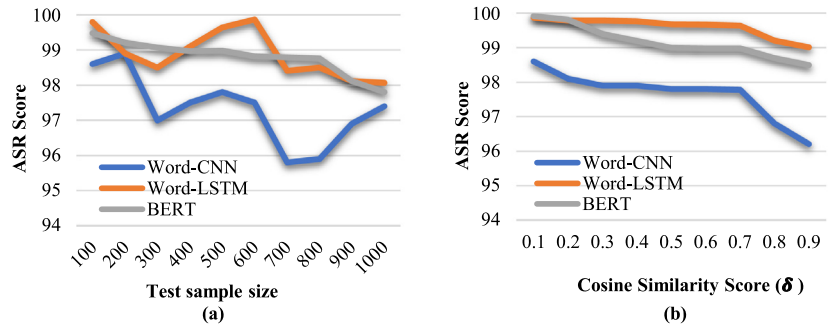


Fig. 9. The fluctuations in the ASR values with the variations in (a) test sample size and (b) cosine similarity score for the models trained on **MR** dataset.

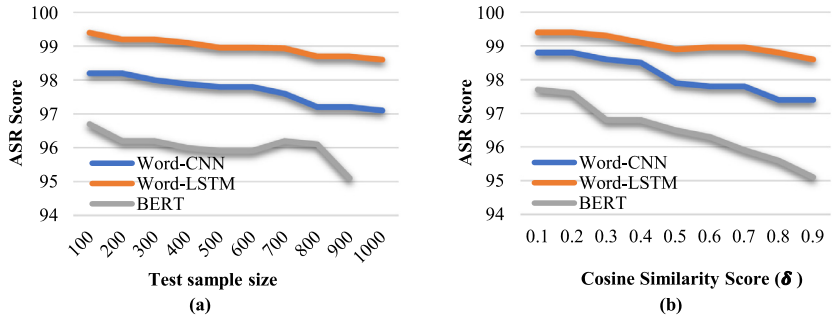


Fig. 10. The fluctuations in the ASR values with the variations in (a) test sample size and (b) cosine similarity score for the models trained on **IMDB** dataset.

C. Scalability: To assess the efficacy of the suggested algorithm, a collection of test specimens has been established within a predetermined range of values that have been subjected to HOMOCHAR-induced perturbations. To evaluate the variability of ASR values in relation to changes in the scale of the test samples, the success rates of three models were examined during the process. As the scope of the test samples was progressively increased throughout this procedure, there was a corresponding increase in the mean runtime required to produce adversarial samples. As illustrated in Figs. 9(a) and 10(a). The observation drawn from the illustration suggests an unfavourable relationship between the population sample size and the ASR scores. This correlation is characterized by a discernible downward trend as the sample size increases. However, it is noteworthy that the ASR scores do not exhibit a huge substantial variation with changes in the sample size.

D. Utility Analysis: It is evident that the adversarial texts produced by HOMOCHAR exhibit a greater degree of similarity to the original texts in comparison to those generated by conventional baseline attack algorithms. From Fig. 3, it can be concluded that the adversarial examples generated through proposed methodology are more effective in preserving utility. The rationale behind this is that the baseline methods encompass a range of linguistic errors such as misspellings, insertions, transpositions, and synonym substitutions etc., The HOMOCHAR algorithm operates by substituting characters in the original review with homoglyph characters, resulting in a perturbed review where each character appears visually identical. This technique is designed to deceive neural text classifiers. Sometimes, the perturbed review does not appear adversarial even to the adversary, this can be observed in all the examples presented in Fig. 3. Homographs exhibit distinct tokenization and are considered "out of vocabulary" in the word embedding domain. Therefore, it can be asserted that perturbation generated from HOMOCHAR can be most crucial technique for conducting malicious manipulation in text classification.

E. Transferability: This investigation examines the transferability of adversarial text, specifically, the extent to which adversarial samples generated from one model can deceive a different model. The present investigation involved the collection of adversarial

Table 10
Transferability of adversarial examples on MR dataset. Row i and column j is the ASR of adversaries generated for model i evaluated on model j.

		Word-CNN	Word-LSTM	BERT
MR	Word-CNN	–	79.1%	89.7%
	Word-LSTM	64.7%	–	82.4%
	BERT	81.2%	81.8%	–

examples from the MR test set that exhibited misclassification by the Word-CNN model. Subsequently, the predictive ASR of the aforementioned examples was assessed in comparison to two additional target models. The findings presented in Table 10 indicate that there exists a moderate level of transferability among the models. Furthermore, the adversarial samples that were produced using the BERT model, which exhibited greater ASR scores, demonstrated a higher degree of transferability.

F. ***Random word perturbation:*** The data presented in Table 11 indicates that the act of randomly selecting words to modify, as denoted by "Random", exhibits minimal impact on the ultimate outcome. This suggests that indiscriminately altering words would not deceive classifiers, and it is imperative to select significant words to modify for a successful attack.

G. Implementation and memory details: The experimentation process was carried out utilizingtwo NVIDIA RTX A5000 GPUs, with a system memory of 128 GB. Equipped with a total of 48 GB of graphics memory, driver version 460.32.03, CUDA version 11.2, and 10 TB GB of hard disc space. The experiments were conducted in a repeated manner, with each experiment being replicated 5 times. The reported value is the mean of the obtained results. The significance of this replication lies in the stochastic nature of training, which results in variability in performance. Stop-words are commonly filtered out during feature extraction in various NLP tasks, including this experiment. This is due to the observation that the presence or absence of stop-words has minimal influence on the outcome of the prediction results. The experiments conducted in this study employ the 200-dimensional GloVe embeddings,[6] which were trained on a corpus of 840 billion tokens sourced from Common Crawl. Moreover, a semantic similarity threshold of 0.7 is established to ensure an optimal balance between the calibre and potency of the produced adversarial text. ***Memory utilization***: During the development of 500 adversarial samples on both datasets, an average of 8.3 GB of RAM, 3.9 GB of graphics memory, and 26.2 GB of disc space were utilized in this investigation.

7. Discussion & future work

NLP models that rely on text as input are vulnerable to a diverse range of subtle perturbations that have the potential to modify model output and prolong inference runtime while leaving the visual appearance of the input unchanged. These attacks consist of arbitrary character substitutions, insertion, deletion, and substitution of essential words with synonyms. Using homoglyphs, this article presents a novel method for fooling neural sentiment classifiers. Although they have occasionally been observed in obfuscating spam and phishing scams detection mechanisms in the past, it seems that the designers of the various NLP systems that are presently being implemented on a large scale have disregarded these concerns entirely. This article explores the phenomenon of adversarial attacks on natural language sentiment classification applications through the utilization of homoglyphs. The experimental findings reveal that HOMOCHAR attack is superior to other conventional methodologies in terms of both success rates and average perturbation. In addition, the user study demonstrated that it was challenging for users to recognize homoglyph adversarial examples as perturbed text.

[6] http://nlp.stanford.edu/projects/glove/.

7.1. Future scope

Extension to Targeted Attack: This paper solely focuses on conducting untargeted attacks, which involve altering the output of the model. However, it is worth noting that HOMOCHAR exhibits the potential for facile customization in the context of targeted attacks, wherein the model can be compelled to produce a predetermined label. The deliberate modification of a model to generate a predetermined outcome, commonly referred to as a targeted attack., involves a modification of the goal function component in the proposed method.

Extension to wider NLP applications: As future work or research opportunities, we will expand the scope of HOMOCHAR attack to distort a variety of NLP applications such as toxic content detection, rumour detection, smishing & phishing detection etc., in addition to sentiment analysis.

Apply HOMOCHAR attack on API platforms: The advent of machine learning has spurred a proliferation of companies offering their own Machine-Learning-as-a-Service (MLaaS) platforms, which are tailored to Deep Learning Text Understanding tasks, such as text classification. The models are deployed on cloud-based servers and user access is limited to utilizing an application programming interface exclusively. In situations where such a setting is present, a perpetrator is devoid of information regarding the structure of the model, its parameters, or the data used for training. Their sole capability lies in querying the target model, with the output being in the form of prediction or probability scores. The HOMOCHAR model, as developed in the present study, has demonstrated efficacy in operating within black-box scenarios. In future research, it may be feasible to carry out a HOMOCHAR attack on these digital platforms. The utilization of adversarial training (Yoo and Qi, 2021) can be viewed as a solitary measure to alleviate the HOMOCHAR attack on digital platforms, as expounded in the point that follows.

Adversarially Robust Generalization: Using an adversarial training technique (Yoo and Qi, 2021; Terzi et al., 2020) (defencive technique), we will also attempt to propose a model that is resistant to all adversarial perturbations in conjunction with HOMOCHAR. The current investigation delves more extensively into the phenomenon of adversarial examples within the framework of text classification, for future work our aim is to strengthen these systems and enhance the accuracy of classification algorithms through the implementation of adversarial training techniques. To ensure the resilience of applications against malevolent manipulations, it is advisable to suggest that all enterprises involved in the creation and distribution of NLP systems incorporate these protective measures.

7.2. Limitation

The findings of this study indicate the presence of adversarial perturbations in natural language. However, the effectiveness of the perturbations could be enhanced by utilizing a more advanced algorithm, such as particle swarm optimization, to identify and modify significant words. This approach has the potential to further improve the outcomes of the proposed attack.

8. Conclusion

The present study aims to investigate the vulnerability of automatic sentiment classification to adversarial attacks. The results unambiguously indicate that sentiment analysis can be disrupted by altering the vocabulary and syntax for machine learning algorithms while preserving semantic equivalence for human evaluators. The present study examines a deficiency in deep learning models utilized for the purpose of sentiment analysis. By taking advantage of this weakness, in this paper, HOMOCHAR, a novel framework designed to generate adversarial text sequences capable of misleading deep learning networks.

Table 11
Comparison of the ASR scores via random selection of words or via words selected by computing the importance score for perturbation.

Model	Dataset	Accuracy	Random		HOMOCHAR (use heuristic score to find important words)	
			Success rate	Perturbed word	Success rate	Perturbed word
Word-CNN	MR	79.4%	18.6%	15%	97.7%	18.45%
	IMDB	86.3%	32.5%	15%	97.82%	09.33%
Word-LSTM	MR	80.7%	21.7%	15%	99.64%	17.11%
	IMDB	88.3%	24.2%	15%	98.96%	13.08%
BERT	MR	87.5%	41.3%	15%	98.98%	15.40%
	IMDB	89.0%	37.6%	15%	95.91%	11.28%

Furthermore, the study also demonstrated a comparative analysis of various sentiment classifiers to determine which model is more susceptible to adversarial perturbations and which is more robust against them. In general, empirical evidence has demonstrated the feasibility of perturbing automatic sentiment prediction models through adversarial modifications. Therefore, it is imperative to prioritize the development of adversarial robust generalizations over standard generalizations in order to advance societal progress. Furthermore, this study advocates for the exploration of models that exhibit higher resilience against adversarial attacks, as opposed to solely relying on higher accuracy scores. The resilience against the attacks can be enhanced through adversarially robust generalizations, aforementioned in the section of future scope of this article.

CRediT authorship contribution statement

Ashish Bajaj: Software, Validation, Investigation, Data curation, Conceptualization, Methodology, Writing – original draft, Visualization. **Dinesh Kumar Vishwakarma:** Formal analysis, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Boucher, N., Shumailov, I., Anderson, R., Papernot, N., 2021. Bad characters: Imperceptible NLP attacks. [Online]. Available: http://arxiv.org/abs/2106.09898.

Cer, D., et al., 2018. Universal sentence encoder. [Online]. Available: http://arxiv.org/abs/1803.11175.

Corazza, M., Menini, S., Cabrio, E., Tonelli, S., Villata, S., 2020. A multilingual evaluation for online hate speech detection. ACM Trans. Internet Technol. 20 (2), http://dx.doi.org/10.1145/3377323.

Dashtipour, K., Gogate, M., Adeel, A., Larijani, H., Hussain, A., 2021. Sentiment analysis of persian movie reviews using deep learning. Entropy 23 (5), 1–16. http://dx.doi.org/10.3390/e23050596.

Derakhshan, A., Beigy, H., 2019. Sentiment analysis on stock social media for stock price movement prediction. Eng. Appl. Artif. Intell. 85, http://dx.doi.org/10.1016/j.engappai.2019.07.002.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL HLT 2019-2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference.

Feng, S., Wallace, E., Grissom, A., Iyyer, M., Rodriguez, P., Boyd-Graber, J., 2018. Pathologies of neural models make interpretations difficult. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018. http://dx.doi.org/10.18653/v1/d18-1407.

Gao, J., Lanchantin, J., Lou Soffa, M., Qi, Y., 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In: Proceedings - 2018 IEEE Symposium on Security and Privacy Workshops, SPW 2018. pp. 1–21. http://dx.doi.org/10.1109/SPW.2018.00016.

Garg, S., Ramakrishnan, G., 2020a. BAE: BERT-based adversarial examples for text classification. In: EMNLP 2020-2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference. http://dx.doi.org/10.18653/v1/2020.emnlp-main.498.

Garg, S., Ramakrishnan, G., 2020b. BAE: BERT-based adversarial examples for text classification. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. EMNLP, pp. 6174–6181.

Goodfellow, I.J., Shlens, J., Szegedy, C., 2015. Explaining and harnessing adversarial examples. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. pp. 6562–6572.

Han, X., Zhang, Y., Wang, W., Wang, B., 2022. Text adversarial attacks and defenses: issues, taxonomy, and perspectives. In: Security and Communication Networks. Hindawi Limited, http://dx.doi.org/10.1155/2022/6458488, 2022.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Iyyer, M., Wieting, J., Gimpel, K., Zettlemoyer, L., 2018. Adversarial example generation with syntactically controlled paraphrase networks. In: NAACL HLT 2018-2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference. http://dx.doi.org/10.18653/v1/n18-1170.

Jia, R., Raghunathan, A., Göksel, K., Liang, P., 2019. Certified robustness to adversarial word substitutions. In: EMNLP-IJCNLP 2019-2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference. http://dx.doi.org/10.18653/v1/d19-1423.

Jin, D., Jin, Z., Zhou, J.T., Szolovits, P., 2019. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 8018–8025, [Online]. Available: http://arxiv.org/abs/1907.11932.

Kim, Y., 2014. Convolutional neural networks for sentence classification. In: EMNLP 2014-2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference. http://dx.doi.org/10.3115/v1/d14-1181.

Kishwar, A., Zafar, A., 2023. Fake news detection on Pakistani news using machine learning and deep learning. Expert Syst. Appl. 211 (2022), 118558. http://dx.doi.org/10.1016/j.eswa.2022.118558.

Kuleshov, V., Thakoor, S., Lau, T., Ermon, S., 2018. Adversarial examples for natural language classification problems. In: ICLR 2018: International Conference on Learning Representations.

Kurakin, A., Goodfellow, I., Bengio, S., 2019. Adversarial examples in the physical world. In: 5th International Conference on Learning Representations, ICLR 2017 - Workshop Track Proceedings. pp. 1–14, [Online]. Available: http://arxiv.org/abs/1607.02533.

Lakshmi Devi, B., Varaswathi Bai, V., Ramasubbareddy, S., Govinda, K., 2020. Sentiment analysis on movie reviews. In: Advances in Intelligent Systems and Computing. Springer, pp. 321–328. http://dx.doi.org/10.1007/978-981-15-0135-7_31.

Lan, Z., et al., 2020. ALBERT: A lite bert for self-supervised learning of language representations. In: International Conference on Learning Representations. ICLR, pp. 1–17, [Online]. Available: https://github.com/google-research/ALBERT.

Li, J., Ji, S., Du, T., Li, B., Wang, T., 2019. TextBugger: Generating adversarial text against real-world applications. In: 26th Annual Network and Distributed System Security Symposium. pp. 1–15. http://dx.doi.org/10.14722/ndss.2019.23138.

Liang, B., Li, H., Su, M., Bian, P., Li, X., Shi, W., 2018. Deep text classification can be fooled. In: IJCAI International Joint Conference on Artificial Intelligence. pp. 4208–4215. http://dx.doi.org/10.24963/ijcai.2018/585.

Liu, Y., et al., 2019. RoBERTa: A robustly optimized BERT pretraining approach. In: International Conference on Learning Representations. ICLR, pp. 1–15, [Online]. Available: http://arxiv.org/abs/1907.11692.

Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C., 2011. Learning word vectors for sentiment analysis. Association for Computational Linguistics, Portland, Oregon, pp. 142–150.

Morris, J.X., Lifland, E., Yoo, J.Y., Qi, Y., 2020. TextAttack: A framework for adversarial attacks in natural language processing. pp. 119–126, ArXiv.

Naber, D., Kummert, P.F., Fakultät, T., Witt, A., 2003. A rule-based style and grammar checker.

Pang, B., Lee, L., 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: ACL-05-43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference.

Park, S., Cho, J., Park, K., Shin, H., 2021. Customer sentiment analysis with more sensibility. Eng. Appl. Artif. Intell. 104, http://dx.doi.org/10.1016/j.engappai.2021.104356.

Pruthi, D., Dhingra, B., Lipton, Z.C., 2020. Combating adversarial misspellings with robust word recognition. In: ACL 2019-57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference. http://dx.doi.org/10.18653/v1/p19-1561.

Qiu, S., Liu, Q., Zhou, S., Huang, W., 2022. Adversarial attack and defense technologies in natural language processing: A survey. Neurocomputing.

Ren, S., Deng, Y., He, K., Che, W., 2020. Generating natural language adversarial examples through probability weighted word saliency. In: ACL 2019-57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference. http://dx.doi.org/10.18653/v1/p19-1103.

Ribeiro, M.T., Wu, T., Guestrin, C., Singh, S., 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In: ACL 2020-58th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers). pp. 4902–4912.

Ryzhova, A., Devyatkin, D., Volkov, S., Budzko, V., 2022. Training multilingual and adversarial attack-robust models for hate detection on social media. In: Procedia Computer Science. Elsevier B.V., pp. 196–202. http://dx.doi.org/10.1016/j.procs.2022.11.056.

Sanh, V., Debut, L., Chaumond, J., Wolf, T., 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. pp. 2–6.

Shamrat, F.M.J.M., et al., 2021. Sentiment analysis on twitter tweets about COVID-19 vaccines using NLP and supervised KNN classification algorithm. Indones. J. Electr. Eng. Comput. Sci. 23 (1), http://dx.doi.org/10.11591/ijeecs.v23.i1.pp463-470.

Shaukat, K., Luo, S., Varadharajan, V., 2022. A novel method for improving the robustness of deep learning-based malware detectors against adversarial attacks. Eng. Appl. Artif. Intell. 116, http://dx.doi.org/10.1016/j.engappai.2022.105461.

Sun, X., Sun, S., 2021. Adversarial robustness and attacks for multi-view deep models. Eng. Appl. Artif. Intell. 97, http://dx.doi.org/10.1016/j.engappai.2020.104085.

Szegedy, C., et al., 2014. Szegedy others intriguing properties of neural networks. In: 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings. pp. 1–10.

Terzi, M., Susto, G.A., Chaudhari, P., 2020. Directional adversarial training for cost sensitive deep learning classification applications. Eng. Appl. Artif. Intell. 91, http://dx.doi.org/10.1016/j.engappai.2020.103550.

Wang, X., Jin, H., Yang, Y., He, K., 2021. Natural language adversarial defense through synonym encoding. In: 37th Conference on Uncertainty in Artificial Intelligence, UAI 2021.

Wang, X., Li, J., Kuang, X., an Tan, Y., Li, J., 2019. The security of machine learning in an adversarial setting: A survey. J. Parallel Distrib. Comput. 130, http://dx.doi.org/10.1016/j.jpdc.2019.03.003.

Wang, W., Wang, R., Wang, L., Wang, Z., Ye, A., 2023. Towards a robust deep neural network against adversarial texts: A survey. IEEE Trans. Knowl. Data Eng. 35 (3), http://dx.doi.org/10.1109/TKDE.2021.3117608.

Wolff, M., 2020. Attacking neural text detectors. pp. 1–8.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V., 2019. XLNet: Generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems.

Yoo, J.Y., Morris, J.X., Lifland, E., Qi, Y., 2020. Searching for a search method: Benchmarking search algorithms for generating NLP adversarial examples. In: Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP. pp. 323–332, [Online]. Available: https://github.com/QData/TextAttack.

Yoo, J.Y., Qi, Y., 2021. Towards improving adversarial training of NLP models. In: Findings of the Association for Computational Linguistics, Findings of ACL: EMNLP 2021. http://dx.doi.org/10.18653/v1/2021.findings-emnlp.81.

Yuan, X., He, P., Zhu, Q., Li, X., 2019. Adversarial examples: Attacks and defenses for deep learning. IEEE Trans. Neural Netw. Learn. Syst. 30 (9), http://dx.doi.org/10.1109/TNNLS.2018.2886017.

Zang, Y., et al., 2020. Word-level textual adversarial attacking as combinatorial optimization. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 6067–6080. http://dx.doi.org/10.18653/v1/2020.acl-main.540.