Neurocomputing 492 (2022) 278–307

Contents lists available at ScienceDirect

Neurocomputing

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

Survey paper

Adversarial attack and defense technologies in natural language processing: A survey

Shilin Qiu [a,*], Qihe Liu [a], Shijie Zhou [a], Wen Huang [b,c]

[a] School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
[b] Chengdu University of Information Technology
[c] HUAWEI TECHNOLOGIES CO., LTD

ARTICLE INFO

Article history:
Received 24 May 2021
Revised 14 March 2022
Accepted 3 April 2022
Available online 7 April 2022
Communicated by Zidong Wang

Keywords:
Textual adversarial example
Adversarial attack
Adversarial defense
Natural language processing
Artificial intelligence

ABSTRACT

Recently, the adversarial attack and defense technology has made remarkable achievements and has been widely applied in the computer vision field, promoting its rapid development in other fields, primarily the natural language processing domain. However, discrete semantic texts bring additional restrictions and challenges to successfully implementing adversarial attacks and defenses. This survey systematically summarizes the current progress of adversarial techniques in the natural language processing field. We first briefly introduce the textual adversarial example's particularity, vectorization, and evaluation metrics. More importantly, we categorize textual adversarial attacks according to the combination of semantic granularity and example generation strategy. Next, we present commonly used datasets and adversarial attack applications in diverse natural language processing tasks. Besides, we classify defense strategies as passive and active methods considering both input data and victim models. Finally, we present several challenging issues and future research directions in this domain.

1. Introduction

With the rapid progress of high-performance computational equipment and the continuous accumulation of massive data, artificial intelligence technology has been greatly developed and widely used in computer vision (CV) [1–3], natural language processing (NLP) [4–6], voice control [7,8], and other tasks [9–11]. Hence, various intelligent systems are extensively applied to communication, transportation, healthcare, public security, and financial transaction in the real world [12–16].

However, Szegedy et al. [17] proposed the concept of *adversarial example* against image classifiers, demonstrating a tremendous security risk in current intelligent systems. They indicated that the adversarial example generated by adding tiny perturbations to the pure image could make a classifier with good performance output a wrong prediction. More noteworthy, deep neural networks trained on different datasets or with distinct structures can produce the same misclassification for the same adversarial example. Since then, adversarial attack technology has become a research highlight in the artificial intelligence security domain. Thus, sign recognition [18], object detection [19,20], audio recognition [21–23], sentiment analysis [24,25], and malware detection systems [26] are apparently vulnerable when facing adversarial threats.

1.1. Development overview of adversarial technology in NLP field

Based on the paper list[1] collected by Carlini, this survey counts the number of publications related to adversarial in the CV and NLP fields. As shown in Fig. 1, adversarial technology has attracted attention and has developed rapidly. Compared with studies in the CV field, the publications in the NLP domain are far less. However, due to the wide application of NLP technology in text classification [27,28], sentiment analysis [29], text question-answering [30], neural machine translation [31], text generation [32,33] and other tasks, as well as the continuous deepening of adversarial attack and defense technologies, the textual adversarial technology has gradually gained researchers' attention.

Papernot et al. [34] is the first to investigate adversarial attacks on texts. Inspired by the idea of generating adversarial images, they crafted adversarial texts through the forward derivative associated with texts' embeddings. Since then, in order to explore the security blind spot in NLP systems and seek corresponding defense strategies, scholars have conducted in-depth research on NLP

* Corresponding author.
E-mail addresses: 742452674@qq.com (S. Qiu), qiheliu@uestc.edu.cn (Q. Liu), sjzhou@uestc.edu.cn (S. Zhou), 562421007@qq.com (W. Huang).

[1] https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html.

https://doi.org/10.1016/j.neucom.2022.04.020
0925-2312/

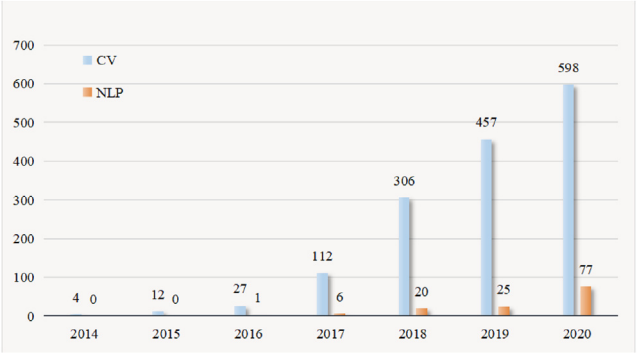

Fig. 1. Publications on adversarial attack and defense technologies by the end of December 2020. The blue represents research in the CV field, while the orange indicates research in the NLP field.

model security from the adversarial attack and defense perspective. Initially, to maintain the semantic consistency and syntactic correctness of texts, researchers transferred adversarial technologies in the image domain to the text domain by introducing specific unique methods, such as the Greedy Search algorithm [35] and the Reinforcement Learning (RL) [36,37]. Later, researchers started to propose special adversarial attacks in the text domain since the text is composed of discrete symbols. In the beginning, works in [38–41] utilized specific techniques to change several characters or words in given texts. These approaches maintain semantic consistency and syntactic correctness well, but the generated adversarial texts generally lack diversity. Latter, researchers [42–44] focused on manipulating the whole text to generate more diverse adversarial texts while maintaining the semantics and syntax of the text. Generally speaking, these methods need to be more carefully designed to ensure the attack ability and the quality of adversarial texts. The research process on defense strategies can be divided into two stages. In the early stage, researchers proposed defense methods based on the model input checking [45,24] and Adversarial Training (AT) [25,35,46,47]. Recently, to alleviate problems like the over-fitting brought by the AT, researchers studied other defense strategies for model reinforcement [41,48,49].

1.2. Comparison of existing relevant surveys

Several researchers have provided different review publications to understand adversarial attack and defense technology. From the perspective of security concerns, Gilmer et al. [50] argued that existing defense approaches have mostly considered abstract toy games that do not relate to any specific security issue. Thus, they discussed and established a taxonomy of motivations, constraints, and abilities for adversaries in all fields. They summarized current attacks from the defender's perspective. To conclude the development and application of adversarial technology in various fields, works in [51–54] elaborated from different perspectives. Among them, Chakraborty et al. [51] focused on three main attack scenarios (the Evasion Attack, Poisoning Attack, and Exploratory Attack) identified by the attack surface. Since adversarial attacks can occur during the training and inference stages of the victim model, Qiu et al. [54] referred to the implementation procedure of adversarial attacks. More importantly, they summarized adversarial attacks' applications in several tasks, such as face recognition, object detection, machine translation, malware detection, and even cloud service.

Focusing on the NLP domain, works in [55–57] have concluded related techniques with various focuses. Specifically, Wang et al. [55] classified adversarial attacks according to NLP tasks. They elaborated on adversarial attacks in text classification tasks but described them simply in other tasks. In addition, they paid attention to the model's robustness and concluded testing and verification methods in the CV and NLP domains. Zhang et al. [56] identified an attack method from two aspects, that is, whether it comes from the CV field and whether it has access to the victim model knowledge. Moreover, they expounded popular NLP models, benchmark datasets, and other relevant information. Since the text is composed of discrete symbols, Huq et al. [57] took the modified component type, which is called semantic granularity, as the basis to classify adversarial attacks. Thus, they divided attack methods into four categories (the character-level, word-level, sentence-level, and multi-level attack).

However, there are still three critical issues in summarizing textual adversarial technologies. The first is, since the text is discrete, classifying adversarial attacks according to semantic granularity is more intuitive than NLP tasks or black/white-box scenarios. The second is that most reviews paid little attention to defense techniques, which are classified and elaborated inadequately. The last is that adversarial technologies' applications in the NLP domain were not elaborated sufficiently. In order To provide readers with a systematic overview of textual adversarial techniques, this survey contributes in three ways, as shown in Fig. 2. Firstly, we classify textual adversarial attacks according to the semantic granularity at the top level and further divide each class into subclasses. Compared with classification methods in [55,56], the proposed approach is more consistent with the NLP scenario. Furthermore, we improve the existing approach [57] by dividing each class at the second level. Secondly, we conclude the defense strategies as passive and active methods, considering both input data and victim model. Compared to methods [55–57] that divided defense strategies into two types (adversarial training and others) directly, the proposed method is more comprehensive and detailed. Finally, we summarize popular benchmark datasets and applications of adversarial technologies in various NLP tasks. In contrast, works in [55,57] provided a brief description of applications. Based on the work [56], we have taken a more systematic look at the application of textual adversarial techniques, including related datasets, victim models, and NLP tasks.

Considering the large number of abbreviations used in this survey, we collected abbreviations frequently used in this work into Table 1 for readers to access the whole work better.

1.3. Explanations for adversarial example's emergence

The effectiveness of adversarial examples is critical to implementing successful adversarial attacks in both CV and NLP domains. Hence, researchers presented several possible explanations for the emergence of adversarial examples. In the early stage, researchers proposed some model-based explanations, including the insufficient generalization ability [67], the extreme nonlinear characteristic of deep neural network (DNN) [17], and the linear behavior in the high-dimensional space [19]. Recently, researchers analyzed the distribution and characteristics of input data. Then, they proposed *Tilted Boundary* hypothesis [68] and the explanation "*Adversarial examples are features*" [69]. Unfortunately, there is still no unified explanation for adversarial examples' emergence.

1.4. Classification of textual adversarial attack

For implementing adversarial attacks on NLP models, Papernot et al. [34] generated adversarial texts by gradient calculation on the embedding space, being the same as the idea of adversarial image generation approaches. Later, researchers proposed diverse adversarial attack methods for continuous embedding and discrete texts. Current textual adversarial attacks can be classified according to various criteria, such as model access, target type, victim DNN type, and semantic granularity. The semantic granularity,

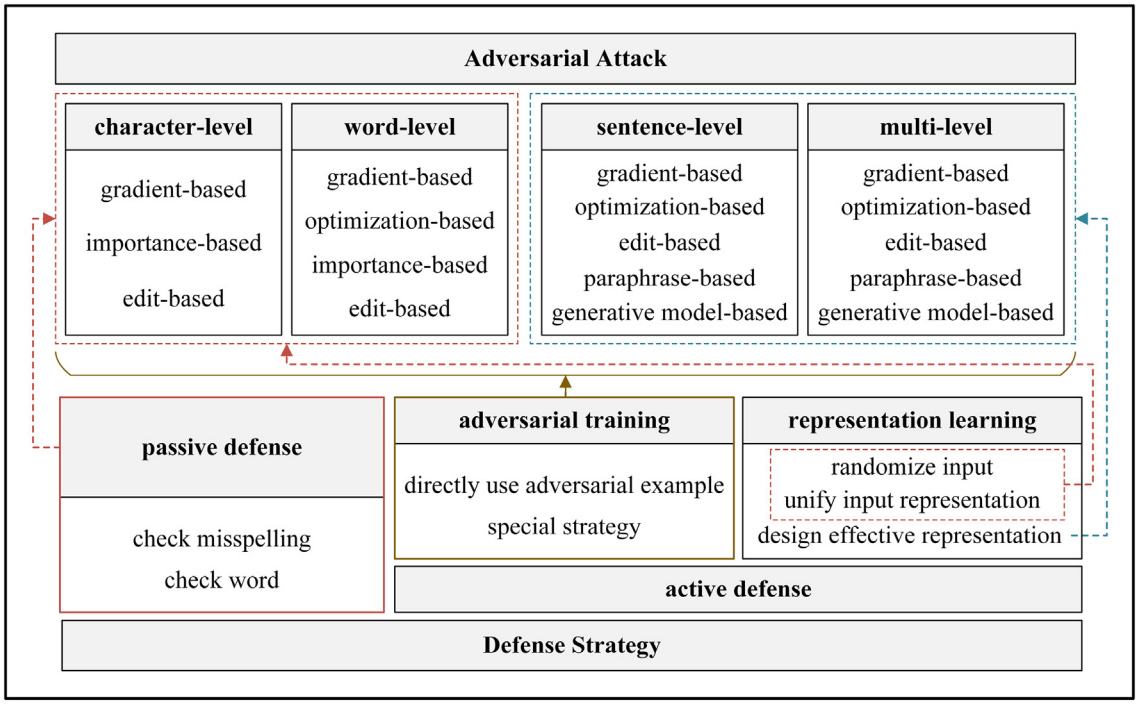

Fig. 2. Summary of the textual adversarial attack and defense strategies in this survey. The attack methods are classified according to the combination of semantic granularity and adversarial example generation strategy. The defense strategies are divided into passive and active methods. Moreover, the attacks concerned by each kind of defense method are marked in the figure.

which refers to the type of modified object, is only appropriate for attacks on texts. Considering that adversarial text generation methods are pretty different, we conclude textual adversarial attack methods comprehensively according to the combination of semantic granularity and example generation strategy. Specifically, we categorize adversarial attacks as four classes (the character-level, word-level, sentence-level, and multi-level method) at the top level according to the semantic granularity. Moreover, we further divide each class into subclasses (the gradient-based, optimization-based, importance-based, edit-based, paraphrase-based, and generative model-based method) according to example generation strategies. To the best of our knowledge, we are the first to propose this kind of two-level classification for adversarial attacks.

In character-level attacks, the adversary inserts, deletes, replaces or swaps characters in a given text. To achieve this purpose, researchers utilized the gradient calculation to rank adversarial manipulations [35] or train a substitute DNN [60]. Besides, *DeepWordBug* [45] used the importance of tokens to determine which character to be changed. The edit-based method in [38,39] used natural and synthetic noises to generate visual and phonetic adversarial examples. Generally speaking, adversarial texts crafted by character-level attacks often have apparent grammar or spelling errors, making it easy for human eyes or misspelling checkers to observe these malicious texts.

In word-level attacks, adversaries modify the word in a given text, generally causing fewer grammar and spelling errors than character-level attacks. The gradient-based methods [34,65,119] utilized the sign of gradient, magnitude of gradient, and gradient itself. Due to the semantic consistency requirement, Michael et al. [70] demonstrated "*adversarial examples should be meaning-preserving on the source side, but meaning-destroying on the target side*". *Seq2Sick* [64] introduced the Group Lasso and Gradient Regularization to a projected gradient method. In contrast, optimization-based, importance-based, and edit-based methods are designed for textual adversarial attacks. Therefore, the generated texts are usually more imperceptible in grammar and syntax. For optimization-based methods, most [40,41,71,72,47,73] are based on Evolutionary Algorithms (such as the Genetic Algorithm (GA) [74] and Particle Swarm Optimization algorithm (PSO) [75]), which do well in handling discrete data but are relatively time-consuming and computationally expensive. Furthermore, researchers proposed specific methods for certain issues, such as the variability exhibited by second-language speakers and first-language dialect speakers [76], the problem of weighing unlikely substitutions high and limiting the accuracy gain [77], and the inapplicability of massive query operations in the real world [37]. For importance-based methods, to craft semantics-preserving texts with minimum modifications, methods in [78–80] ranked words according to the class probability changes obtained by removing words one by one. Among them, the method in [78] works best for the dataset that has sub-categories within each class. Besides, Ren et al. [81] determined the word replacement order by both the word saliency and classification probability, performing well in effectively reducing substitutes. Additionally, Hossam et al. [82] employed cross-domain interpretability to learn the word importance to handle the issue of computational complexity and query consumption. Yang et al.[83] proposed a systematic probabilistic framework, which does well in achieving a high success rate and efficiency. Overall, these importance-based methods are more efficient than optimization-based methods. For the edit-based method, Zhang et al. [84] employed a Metropolis-Hastings sampling approach to replace or randomize words, Li et al. [85] introduced a pre-trained masked language model, Emelin et al. [86] detected word sense disambiguation biases. Overall, word-level attacks usually do better in maintaining semantic consistency and imperceptibility than other attacks, but their generated adversarial examples are less varied.

In sentence-level attacks, the adversary inserts new sentences, replaces words with their paraphrases, or even changes the structure of original sentences. Adversaries generally crafted the universal adversarial perturbation through the iterative projected

Table 1
Abbreviation List.

Abbreviation	Explanation
ADDANY	method proposed in [58]
ADDANY-KBEST	method proposed in [59]
ADDSENT	method proposed in [58]
AdvExpander	method proposed in [42]
AT	Adversarial Training
BERT	Bidirectional Encoder Representations from Transformers
BiDAF	Bi-Directional Attention Flow network
BiLSTM	bidirectional Long Short Term Memory Network
CAT-Gen	*Controlled Adversarial Text Generation*
CharCNN	character-level Convolutional Neural Network
CharCNN-LSTM	character-level Convolutional Neural - Long Short Term Memory Network
CharLSTM	character-level Long Short Term Memory Network
CNN	Convolutional Neural Network
CoCoA	Collaborative Communicating Agents dataset
ConvS2S	Convolutional Sequence to Sequence network
CRNN	Convolutional Recurrent Neural Network
CV	computer vision
DeepWordBug	method proposed in [45]
DISTFLIP	method proposed in [60]
DNN	deep neural network
Doc2Vec	method proposed in [61]
ELMo	Embeddings from Language Models
FGSM	Fast Gradient Sign Method
GA	Genetic Algorithm
GAN	Generative Adversarial Network
GPT-2	language model proposed in [62]
GRU	Gate Recurrent Unit
HotFlip	method proposed in [25]
IMDB	Internet Movie Database
LSTM	Long Short Term Memory Network
MHA	Metropolis Hastings Attack
NLP	natural language processing
MPQA	Multi-Perspective Question Answering
MR	Rotten Tomatoes Movie Review
MSCOCO	Microsoft COCO dataset
OOV	out-of-vocabulary
Paragraph2Vec	method proposed in [61]
PSO	Particle Swarm Optimization algorithm
RewritingSampler	method proposed in [43]
RL	Reinforcement Learning
SCPNs	*Synthetically Controlled Paraphrase Networks* [63]
Seq2Sick	method proposed in [64]
SNLI	Stanford Natural Language Inference Corpus
SQuAD	Stanford Question Answering Dataset
SST	Stanford Sentiment Treebank
TextBugger	method proposed in [24]
TextFool	method proposed in [65]
VAT	Virtual Adversarial Training
VQA	Visual Question Answering Dataset
WMD	Word Mover's Distance
Word2Vec	method proposed in [66]
WordCNN	word-level Convolutional Neural Network
WordLSTM	word-level Long Short Term Memory Network

gradient-based approach on embedding space [87], the projected gradient descent with l_2 regularization [88], and the *HotFlip* [25] based gradient guided approach [89,90]. The method in [88] does well in avoiding the out-of-distribution noise vector and maintaining the naturalness of generated texts. The work in [90] adopted a conditional language model and thus effectively crafted semantically valid statements containing at least one trigger. For the edit-based method, Jia et al. [58] crafted arbitrary sequences of English words in *ADDANY* and sentences that look similar to the question in *ADDSENT*, and Wang et al. [46] improved *ADDSENT* by adding semantic relationship features, Nizar et al. [59] proposed a variant of *ADDANY*. Some of these edit-based methods generate adversarial triggers that do not affect text semantics, except for the readability. For the paraphrase-based method, researchers [91,63] regarded the paraphrase of a given text as its adversarial example. Among them, *Synthetically Controlled Paraphrase Networks* (*SCPNs*) [63] is an encoder-decoder based framework with soft attention [92] and copy mechanism [93], aiming to craft a paraphrase with specified syntax. Although it uses the target strategy, it does not specify the target output. Besides, *AdvExpander* [42] is the first to generate an adversarial text by expanding the original sentence. It differs from existing substitution-based methods and introduces rich linguistic variations to adversarial texts. For the generative model-based method, works in [94,36] leveraged the Generative Adversarial Network (GAN) [95] to craft adversarial examples, with the help of a converter and an autoencoder, respectively. The work in [36] utilized the RL [96] to guide the training of GAN. Differently, the *Controlled Adversarial Text Generation* (*CAT-Gen*) [97] generates adversarial texts by using the controllable attribute that is known to be irrelevant to the task label. Generally, adversarial examples generated by these sentence-level methods are semantics-preserving and full of diversity. However, some of them have reduced readability caused by adding meaningless token sequences.

In multi-level attacks, the modified objects could be two or three kinds of character, word and sentence. The gradient-based method *HotFlip* [25] modified characters and words in a given text by the gradient computation on the one-hot representation. However, it is unsuitable for large-scale attacks since it only generates a few successful adversarial examples under strict constraints. The optimization-based *RewritingSampler* [43] efficiently rewrites the original sentence by word and sentence-level changes, with the help of the word embedding sum and *GPT-2* language model [62]. The importance-based *TextBugger* [24] used the Jacobian matrix to calculate each word's importance in white-box settings and determined the split sentence importance by querying the victim model in black-box settings. However, in *TextBugger*, only a few editing operations can mislead the classifier. Unlike most works focusing on semantic tasks, Zheng et al. [44] is the first to construct a syntactic adversarial example at both sentence and phrase levels. For the edit-based method, Niu et al. [98] proposed *Should-Not-Change* and *Should-Change* adversarial strategies, Blohm et al. [99] implemented four adversarial text generation methods with diverse edits on the word and sentence level. The hybrid encoder-decoder-based *Adversarial Examples Generator* [100] introduced both word and character-level perturbations with the help of a self-critical approach. In general, adversarial examples generated by these multi-level methods are more varied than those crafted by the character and word-level methods to a certain extent. However, these attack methods always have more constraints.

1.5. Applications of textual adversarial attack

To explore potential adversarial attack threats faced by existing NLP intelligent systems and further provide a foundation for developing effective defense strategies, researchers have applied various adversarial attacks in extensive NLP tasks, such as text classification, neural machine translation, and dialogue generation. Considering that these NLP tasks are exceedingly different from each other and performed on distinct datasets, we summarize well-known benchmark datasets used in NLP tasks. Then, we give a brief introduction about the applicable task, dataset size, and characteristics of the principal datasets (see details in Section 4.1). After that, we classify existing adversarial attack applications into eight categories according to text processing. Besides, we elaborate on the application scenarios, victim models, benchmark datasets, and attack effects of current adversarial attack methods applied in each kind of NLP task (see details in Section 4.2). Note that most attack methods are simultaneously applied in multiple tasks, indicating the transferability of these attack methods across datasets, DNN models, and even NLP tasks.

1.6. Defenses against textual adversarial attack

Due to the deepening development of adversarial attack technologies and the wide application of textual adversarial attacks, researchers have realized the severity of adversarial attack threats in NLP systems, promoting various defense methods. This survey categorizes current defense strategies as passive and active defenses. The most commonly used passive defense method is checking misspellings and typos in the textual input [45,24,101–103]. However, these passive strategies are suitable for defending against character and word-level attacks. The active defenses principally contain adversarial training and representation learning strategies. For the adversarial training, some researchers [25,35,24,58,46,38,47] directly added adversarial examples generated by existing attacks to the training dataset. Considering the massive calculation consumption and low efficiency of these above methods, researchers [104,105] proposed GAN-style approaches to train NLP models together with the adversarial example generator. The work in [106] presented a variation of Virtual Adversarial Training (VAT) to generate virtual adversarial examples and virtual labels in the embedding space. Additionally, researchers have paid attention to the issues of out-of-vocabulary (*OOV*) words [107], distribution difference [107], diversified adversarial example requirement [108], and offensive language detection in the real world [109]. Although adversarial training can effectively improve the robustness of NLP models and overcome problems like adversarial example preparation and calculation consumption, it is likely to reduce their classification accuracy. For the representation learning, researchers improved the input representation ability of NLP models. Some methods [110–112] introduced random perturbations to the input during the training step. Works in [38,41,48,49] improved the generalization of models by encoding input and their neighbors with the same representation. Other researchers have designed more effective representation approaches by fine-tuning both local and global features [113], introducing disentangled representation learning [114], linking the multi-head attention to structured embedding [115], and augmenting input sentences with their corresponding predicate-argument structures [116]. In general, randomizing input and unifying input representation is similar to the passive defense, and by contrast, designing effective representation is more challenging. Furthermore, current defense research is much less than attack research, which reminds researchers that it is necessary to pay more attention to defense strategies against adversarial attacks.

1.7. Contribution and organization

Our contributions. This survey concentrates on the adversarial attack and defense technology in the NLP field and provides a thorough and systematic review. The key contributions of this survey can be summarized as follows:

- We comprehensively and systematically summarize the textual adversarial attack and defense technology, elaborating on textual adversarial examples, adversarial attacks on texts, defenses against textual adversarial attacks, applications in various NLP tasks, and potential development directions in this domain.
- We categorize current textual adversarial attacks according to the semantic granularity at the top level and further classify each class into several subclasses depending on the example generation strategy. To the best of our knowledge, we are the first to regard the example generation strategy as a classification criterion and propose this two-level classification for adversarial attacks.
- We classify existing defense strategies against textual adversarial attacks as passive and active methods, simultaneously considering the victim model stages and starting points of defense strategies. Compared with existing surveys, our classification method is the only one that considers both the input data and the victim model.

Organization. In Section 2, we introduce causes of the adversarial example's emergence, particularities of adversarial texts, vectorization of textual data, and evaluation metrics of adversarial texts. In Section 3, we classify textual adversarial attack methods from four aspects and elaborate them according to the combination of semantic granularity and example generation strategy. In Section 4, we present benchmark datasets and applications of textual adversarial attacks. In Section 5, we summarize existing defense methods against adversarial attacks on texts. Finally, we conclude several significant challenges and potential development directions in Section 6.

2. Textual adversarial example

Szegedy et al. [17] proposed that adversarial images, generated by adding tiny noises to pixel values, can easily make an image classifier produce wrong predictions but do not affect the perception of human eyes. It is the first study on adversarial attack technology in the CV domain. In the NLP field, Papernot et al. [34] is the first to investigate adversarial attacks. They crafted adversarial texts invisible to human eyes by modifying characters or words in original texts. As shown in Fig. 3, a text correctly categorized by a sentiment analysis model is classified wrongly after replacing one word of it.

To let readers have a more precise and comprehensive understanding of the textual adversarial example, we first present possible reasons for the emergence of adversarial examples in Section 2.1. Then, we conclude the particularities of textual adversarial examples in Section 2.2. Later, we elaborate on the vectorization methods for discrete data in Section 2.3. Finally, we summarize the commonly used metrics for evaluating the effectiveness and imperceptibility of adversarial texts in Section 2.4. Note that the particularities described in this survey are unique to textual examples, and the general characteristics of adversarial examples in all fields are elaborated in detail in the literature [54].

2.1. Causes of adversarial example

Realizing the terrifying attack capability of adversarial examples, researchers have begun to explore the reasons for the existence of adversarial examples. Although researchers have presented some assumptions, there is still no widely accepted explanation. In the early stage, researchers thought that the cause was the insufficient generalization ability of DNN to predict unknown data due to over-fitting or inapposite regularization [67]. While Szegedy et al. [17] proposed that the extreme nonlinear characteristic of DNN leads to the existence of adversarial examples.

Later, Goodfellow et al. [19] verified the above two hypotheses. They input adversarial examples into a regularized DNN and discovered that the model's effectiveness for resisting adversarial examples was not significantly improved. Besides, by adding tiny perturbations to the input of a linear model, they showed that if the input dimension is sufficient, it is possible to construct adversarial examples with high confidence. Therefore, they proposed that the linear behavior of DNN in high-dimensional space leads to the emergence of adversarial examples. In other words, since

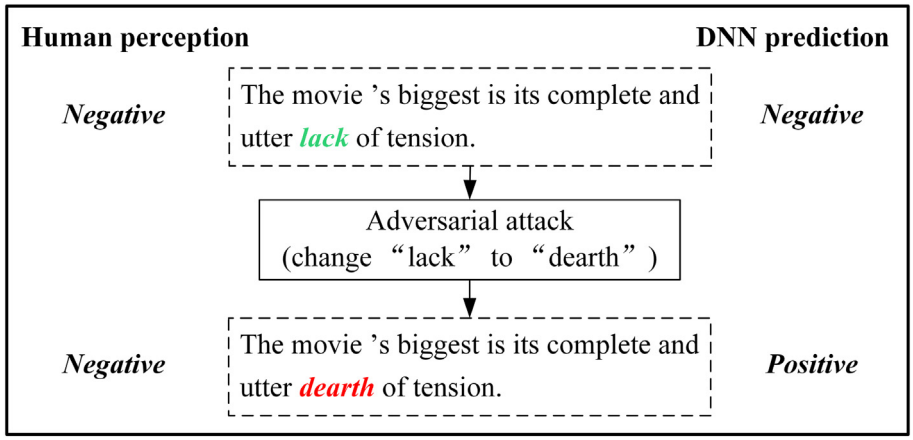

Fig. 3. An instance of adversarial example. Before the attack, the sentiment analysis model classified the original text into a negative class, in line with human judgment. After replacing the word "*lack*" with "*dearth*", the same model produces a wrong prediction.

the nonlinear components of DNN are linear in segments, if adding tiny noise to the input, this noise would persist in the same direction and accumulate at the end of DNN, finally leading to a different output.

Differently, some hypotheses were proposed from the perspective of data characteristics instead of neural network structures. Tanay et al. [68] proposed a hypothesis called *tilted boundary*. They demonstrated that since a DNN is usually unable to fit all data accurately, there is a space for adversarial examples near the classification boundary of the DNN. Ilyas et al. [69] proposed "*Adversarial examples are features*". They thought that the destructiveness of adversarial examples directly results from the model's sensitivity to input features. Hence, they concluded that adversarial examples are not bugs but features that indicate how DNN visualizes everything. Furthermore, they divided input features into robust and non-robust types. They demonstrated that adding tiny perturbations to non-robust features could easily make DNNs produce incorrect outputs.

2.2. Particularities of adversarial text

As mentioned before, publications related to adversarial technology in the NLP field are far less than those in the CV field. The reason is that three extra constraints need to be considered when generating textual adversarial examples. Specifically:

- **Discrete.** Unlike images represented by continuous pixel values, the symbolic text is discrete. Therefore, finding appropriate perturbations is critical to efficient textual adversarial example generation. Researchers proposed two solutions for handling this issue. One is vectorizing discrete texts into continuous representations at first (see specific methods in Section 2.3), and then using perturbation generation approaches for images to craft adversarial texts. The other is carefully designing perturbations directly on the discrete space.
- **Perceivable.** The well-performed adversarial image generation method is based on the premise that a few pixel value changes in an image are invisible to human eyes. However, a slight modification of a character or word is easily realized by human eyes and spelling checkers. Hence, finding textual adversarial examples that are hard to be observed by human eyes is vital for successful adversarial attacks.
- **Semantic.** Compared with images whose overall semantics do not change when changing a few pixel values, the semantics of a text could be altered by even replacing or adding a character, violating the principle that adversarial examples are perceivable to humans. Therefore, keeping the semantics consistent is the key to crafting influential adversarial texts.

2.3. Vectorization of discrete data

Due to additional constraints like grammatical legitimacy and semantic consistency need to be taken into account when designing adversarial example generation algorithms directly on discrete texts, researchers proposed to turn original texts into continuous vectors at first, and then apply methods designed for images or design new algorithms on vectors to craft adversarial ones. Current text vectorization methods mainly include:

- ***One-Hot* encoding.** The *One-Hot* encoding represents a character or a word by a vector, in which only one element is one and all other elements are zero. When mapping a text to a vector x, the length of x is equal to the size of the vocabulary, and the element set to one is determined by its position in the vocabulary. Although this method is simple to implement, the *One-Hot* vector usually is sparse, semantically independent, and with a very high dimension.
- **Word-count-based encoding.** This approach initializes a zero-coded vector with the length of the vocabulary size and then replaces each element with a specific value. The Bag-of-Words ignores the word order, grammar, and syntax of texts, and regards the text as a collection of words. Thus, the Bag-of-Words sets each element of the initialized vector with the corresponding word count. The Term Frequency-Inverse Document Frequency considers the word importance determined by frequencies of the word's appearance in the text and corpus. Like the *One-Hot* encoding, the vector obtained by word-count-based encoding is sparse and semantically independent, but with relatively low dimensions.
- ***N-gram* encoding.** The *N-gram* language model predicts the next likely word when given a text. This is based on the assumption that the occurrence of n-th word is only related to the first $n-1$ words. When $n = 1, 2, 3$, it is called unigram, bigram, and trigram, respectively. The *N-gram* takes the word order into account, but with the increase of n, the vocabulary expands rapidly, and the vector becomes sparse.
- **Dense encoding.** The dense encoding provides a low-dimensional and distributed vector representation for discrete data. The *Word2Vec* [66] uses continuous bag-of-words and skip-gram models to produce a dense representation of words,

called word embedding. Its basic assumption is that words appearing in similar contexts have similar meanings. Thus, the *Word2Vec* alleviates the discreteness and data-sparsity problems to some extent. Similarly, the *Doc2Vec* and *Paragraph2Vec* [61], two extensions of word embedding, encode sentences and paragraphs to dense vectors.

2.4. Evaluations of adversarial text

It is necessary to evaluate the effectiveness of adversarial examples in all fields. Additionally, the imperceptibility estimation of adversarial text is particularly significant for textual adversarial attacks because of their unique characteristics. Consequently, this section outlines the effectiveness and imperceptibility evaluation metrics. The first fits all kinds of adversarial examples, while the last is appropriate for adversarial texts.

2.4.1. Metrics of effectiveness evaluation

The effectiveness evaluation reflects the adversarial examples' ability to make DNN produce wrong predictions. Existing effectiveness evaluation metrics principally include accuracy, error, and accuracy reduction.

- **Accuracy rate.** It refers to the proportion of examples correctly classified by the victim model in total inputs. The lower the accuracy rate is, the more effective adversarial examples are. Otherwise, it means the effectiveness of adversarial examples cannot meet a specific requirement.
- **Error rate.** Opposite to the accuracy rate, the error rate refers to the proportion of examples wrongly classified by the victim model in total inputs. The higher the error rate is, the higher the validity of the adversarial example is.
- **Accuracy reduction.** It is a more intuitive evaluation metric, which refers to the accuracy changes before and after the adversarial attack. The larger the accuracy reduction is, the more effective adversarial samples are.

It is worth noting that these effectiveness evaluation metrics apply in both adversarial attack and defense procedures.

2.4.2. Metrics of imperceptibility evaluation

The fundamental constraint of adversarial examples is that the perturbation must be invisible to human eyes. Hence, adversarial texts demand grammatically correct, syntactic correct, and semantically consistent with the original ones. Since discrete texts can be transformed into continuous numerical vectors through various vectorization approaches, this survey classified imperceptibility evaluation metrics for adversarial text into continuous space and discrete space measures.

Evaluation Metrics on Continuous Space: suitable for vectorized data. The vectorized adversarial data can be evaluated by metrics designed for image data, such as Euclidean Distance. Besides, researchers proposed several particular metrics for assessing adversarial texts, including Cosine Distance and Word Mover's Distance. The Euclidean Distance and Cosine Distance are metrics that measure the semantic similarity between the adversarial text and the original one.

- **Euclidean Distance.** It is also called *L2* distance or *L2* norm. For two given word vectors $\vec{m} = (m_1, m_2, \cdots, m_k)$ and $\vec{n} = (n_1, n_2, \cdots, n_k)$, the Euclidean Distance is expressed as:

$$D(\vec{m}, \vec{n}) = \sqrt{(m_1 - n_1)^2 + \ldots + (m_k - n_k)^2} \tag{1}$$

where k is the dimension of the word vector, m_i and n_i are the i-th factors of $\vec{m}$ and $\vec{n}$. The smaller the Euclidean Distance is, the more similar these two vectors are.

- **Cosine Distance.** It is used to calculate the semantic similarity between two vectors. Compared with the Euclidean Distance, the Cosine Distance is more concerned with the difference of directions between two vectors. For two given word vectors $\vec{m}$ and $\vec{n}$, the cosine similarity is expressed as:

$$D(\vec{m}, \vec{n}) = \frac{\vec{m} \cdot \vec{n}}{\|m\| \cdot \|n\|} = \frac{\sum_{i=1}^{k} m_i \times n_i}{\sqrt{\sum_{i=1}^{k} (m_i)^2} \times \sqrt{\sum_{i=1}^{k} (n_i)^2}} \tag{2}$$

- **Word Mover's Distance** (*WMD*) [117]. It is a variant of the Earth Mover's Distance. It measures the minimum distance that words of one document need to travel to reach words of another document on embedding space. The lower the *WMD* value is, the more similar the two texts are. For two given texts, the *WMD* can be formalized with a minimization problem as follows:

$$\begin{aligned} & min \sum_{i,j=1}^{p} T_{ij} \left\| \vec{m}_i - \vec{n}_j \right\|_2 \\ & \text{s.t.,} \sum_{j=1}^{k} T_{ij} = d_i, \forall i \in \{i, \cdots, k\}, \\ & \sum_{i=1}^{k} T_{ij} = d'_i, \forall j \in \{i, \cdots, k\} \end{aligned} \tag{3}$$

Here, $\vec{m}_i$ and $\vec{n}_j$ are word embeddings of word i and word j, respectively. k is the number of words in the text. d and d' are normalized bag-of-words vectors of two documents, respectively.

Evaluation Metrics on Discrete Space: suitable for textual data without converting texts into vectors in advance. Such metrics mainly include the Jaccard Similarity Coefficient, grammatical and syntactic similarity, Edit Distance, and changes count.

- **Jaccard Similarity Coefficient.** It measures the similarity of finite sets by the intersection and union of sets. Given set A and B, their Jacobian Similarity Coefficient is expressed as:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{4}$$

Where $0 \leqslant J(A, B) \leqslant 1$. In texts, A and B represent two sentences or documents. $|A \cap B|$ denotes the number of similar words in A and B in total, while $|A \cup B|$ represents the number of unique words in total. The closer $J(A, B)$ is to 1, the more similar the two texts are.

- **Grammatical and syntactic similarity.** Researchers proposed three types of evaluation methods for evaluating the grammatical and syntactic correctness and similarity of adversarial texts. One uses the grammar and syntax checker to guarantee that the generated adversarial texts are valid; the second uses the Complexity [118] to assess the quality of a language model for generating adversarial texts; the last ensures the effectiveness of paraphrases in the generation process.
- **Edit Distance.** It measures minimal changes by converting one string to another. The smaller the Edit Distance is, the more similar the two strings are. Levenshtein Distance is known as Edit Distance with insert, delete and replace operations.
- **Count of changes.** It is a straightforward approach that counts the number of modified words or characters in adversarial texts under the word-level model or character-level model, respectively.

3. Textual adversarial attack

Inspired by adversarial image generation methods, Papernot et al. [34] crafted adversarial texts through gradient information on embedding space. Their approach consists of two steps: vectorizing the text and calculating the gradient on continuous space. Based on this idea, several homologous methods like [65,78,25,35,119] are proposed. In contrast, some researchers [38,24,45,60,81,82,72] suggested crafting adversarial texts directly on the discrete space.

The above classification of adversarial attacks depends on whether the modification operation performs on the continuous or discrete space. In fact, adversarial attacks can be categorized according to different criteria, such as model access and target type. To comprehensively review current textual adversarial attacks, we firstly discuss the classification of adversarial attacks from multiple perspectives in Section 3.1. Then, we elaborate on four types of attacks categorized by the modified object type in Section 3.2–3.5.

3.1. Categorization of attack method

This survey applies the three adversarial attack classification criteria presented in [56]: model access, target type, and semantic granularity, as shown in Fig. 4. Furthermore, considering that existing adversarial attacks employed various approaches to craft high-quality adversarial texts, we propose classifying these textual adversarial attack methods according to the example generation strategy. As far as we know, this classification criterion is proposed for the first. The model access and target type can be employed to classify attacks in all fields; however, the other two, especially the semantic granularity, are only suitable for categorizing attack methods in the NLP domain.

3.1.1. Model access

According to the victim model knowledge the adversary can access, adversarial attacks are classified into white and black-box attacks. This classification is the most commonly used in various fields, including the CV and NLP domains. Since this survey does not elaborate on current adversarial attacks according to this criterion, we briefly describe them below.

- **White-box attack.** In white-box scenarios, the adversary has full access to all knowledge about the victim DNN, including its input, output, architecture, parameters, activation functions, and loss functions. Utilizing this available information, adversaries can identify the most vulnerable feature space of the victim model and craft efficient adversarial examples. For example, works in [34,65,119] generated adversarial examples by utilizing the gradient's sign, the gradient's magnitude, and gradient itself, respectively. Furthermore, *TextBugger* [24] firstly uses the Jacobian matrix J to calculate the importance of each word and then uses five editing strategies to modify the text. Blohm et al. [99] leveraged the victim model's sentence-level attention distribution to find the plot sentence and change words receiving the most attention in the plot sentence. Therefore, white-box attacks generally have a high success rate and are a potent threat, and are usually the worst-case attack against a specific model and input data.
- **Black-box attack.** Unlike white-box attacks, the adversary in black-box scenarios only has access to the output by inputting examples to the victim model, but not any other information about the model architecture and parameters. For example, to identify meaningful words in given texts, works in [78,81,79] computed the probability change value, word saliency, and classification probability by using the victim model's output. Researchers proposed several randomized attack methods for extreme conditions where the attacker cannot obtain valuable output from victim models. For instance, *ADDONESENT* [58] adds random sentences recognized by human beings to the end of pure examples. Niu et al. [98] proposed a *Should-Not-Change* strategy with four edits, among which there are two random approaches (randomly transposing neighboring tokens and randomly removing stopwords). It can be seen that a black-box attack's success rate and threat power are generally lower than those of a white-box attack. However, due to no requirement for complete knowledge, black-box attacks are more practical in the real world.

3.1.2. Target type

According to the attack purpose of an adversary, adversarial attack methods are categorized into targeted and non-targeted attacks. Similar to the first classification method, we briefly outline it below.

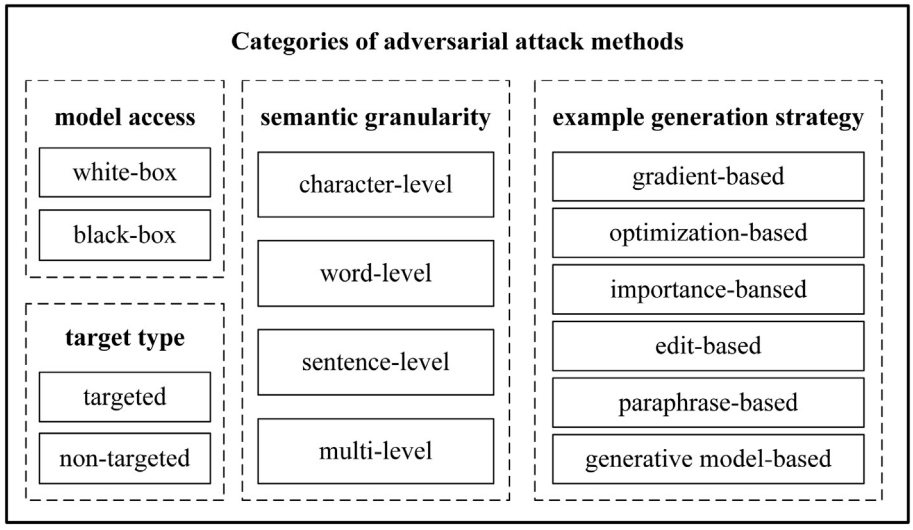

Fig. 4. Categorization of textual adversarial attack methods. There are four criteria: model access, target type, semantic granularity, and example generation strategy.

- **Non-targeted attack.** The adversary hopes to generate an adversarial example x' that makes the victim model f produce a wrong output $f(x') \neq y$, where y is the correct label of the input x. Since there is no limit on the target wrong output, this kind of attack is more frequently employed, such as attacks in [39,119,40].
- **Targeted attack.** In this scenario, the adversary intends to generate adversarial examples that make victim models output a specified wrong prediction. More specifically, the adversary hopes that the generated example x' to cause the victim model f outputting $t = f(x')$, where t is the output specified by the adversary. For instance, Ebrahimi et al. [35] considered that some corruptions of the neural machine translation model's output might be much worse than others, such as translating "*good morning*" as "*attack them*" is much worse than translating it as "fish bicycle". Therefore, they proposed a *controlled attack* that aims to remove a specific word from the translation, and a *targeted attack* that tries to mute a specific word and replace it with another. Thus, targeted attacks are more aggressive on deep neural networks than non-targeted ones. However, they are more challenging to implement since they contain more constraints than non-targeted ones.

3.1.3. Semantic granularity

Considering the type of objects modified by the adversary, textual adversarial attacks can be divided into four types: character-level, word-level, sentence-level, and multi-level attack. The character-level attack refers to inserting, removing, flipping, swapping, or replacing characters in a text, and the word-level attack performs the same operations for words. The sentence-level attack usually inserts extra distractor sentences, generates the paraphrase, or modifies the original sentence structure in the case of semantic consistency. Besides, the multi-level attack simultaneously employs two or three character-level, word-level, and sentence-level attacks.

3.1.4. Example generation strategy

According to different strategies in the adversarial example generation process, we divide adversarial attacks into six types: gradient-based, optimization-based, importance-based, edit-based, paraphrase-based, and generative model-based methods. Among them, strategies like the gradient-based method are evolved from adversarial image generation methods, and the implementation process of these attacks is usually relatively straightforward. While other methods like the optimization-based and edit-based methods are proposed for discrete data, they generally show better performance in maintaining semantic consistency and grammatical correctness; however, they have enormous difficulty when designing well-turned algorithms.

- **Gradient-based.** These methods calculate the forward derivative to the input and obtain adversarial perturbations by gradient backpropagation. Therefore, the vectorization for text needs to be implemented at first. Besides, spelling and grammatical errors commonly exist in generated texts, causing the adversarial example to be perceived.
- **Optimization-based.** It regards the adversarial example generation as a minimax optimization problem, i.e., maximizing the victim model's prediction error while the difference between the adversarial example and the original one is within an acceptable range. Currently, researchers craft adversarial texts essentially based on evolutionary algorithms, such as the GA [74] and PSO [75].
- **Importance-based.** This means that which object is to be modified and how to modify it are determined by each object's importance related to the victim model. Since the more critical the changed word is, the easier it is to change the prediction of the victim model, even if the change is small enough. The adversarial example generated by this strategy generally maintains semantic consistency, grammatical, and syntactic correctness well.
- **Edit-based.** It crafts adversarial examples by operations like inserting, removing, and swapping characters, words, or sentences. These editing operations are also used in other approaches, such as gradient-based, optimization-based, and importance-based methods. Therefore, the edit-based method in this survey refers to attacks that utilize the above editing operations but do not use the gradient information, optimization algorithm, or item importance.
- **Paraphrase-based.** The adversary takes the paraphrase of a sentence as its adversarial example. In the paraphrase generation process, the adversary introduces different extra conditions to fool the victim model without affecting human judgment. The sentence-level attacks commonly use these approaches.
- **Generative model-based.** This method uses the generative model like the GAN [95] and encoder-decoder model to generate adversarial texts, and is frequently used in sentence-level attacks. Since there are gradient back propagations when training the generative model or crafting adversarial examples, these methods are usually combined with other techniques, such as RL [96].

Since the semantic granularity is unique for textual adversarial attack classification, and the example generation strategy indicates the principal idea of generating adversarial examples, we suggest a novel two-level classification method for the textual adversarial attack taxonomy. We first categorize adversarial attack methods as four classes according to the semantic granularity at the top level. Then, we subdivide each kind of method according to the example generation strategy. Section 3.2–3.5 shows details.

3.2. Character-level attack

As mentioned before, the main idea of character-level attacks is to insert, delete, flip, replace, or swap individual characters, special symbols, or figures in a text. These attacks generally use three example generation strategies: gradient-based, importance-based, and edit-based methods, as summarized in Table 2. Adversarial texts crafted by these methods often have apparent grammar or spelling errors, making it easy for human eyes or misspelling checkers to observe these malicious texts.

3.2.1. Gradient-based method

The white-box adversary in [35] calculated the gradient information to rank adversarial manipulations and used the greedy search and beam search method to find adversarial examples. Furthermore, it forced the adversary to remove or change a specific word in a translation by two novel loss functions. In contrast, the black-box adversary in [35] just randomly selected characters and made suitable modifications with them. Since the implicit knowledge in an optimization process can be refined into another more efficient neural network, *DISTFLIP* [60] converts a white-box attack into a black-box one by training a neural network that imitates the adversarial example generation process of *HotFlip* [25]. Due to the independence of optimization algorithms, *DISTFLIP* is ten times faster than *HotFlip* when crafting adversarial texts. How-

Table 2
Summary of Character-level Adversarial Attacks in NLP.

Strategy	Work	White/ Black-box	Targeted/Non-targeted	Main Ideas
gradient-based	[35]	both	both	Using backward gradient propagation to rank adversarial manipulations, and utilizing the greedy search and beam search to seek adversarial examples.
	[60]	black-box	non-targeted	Distilling knowledge of white-box attacks.
importance-based	[45]	black-box	non-targeted	Determining top critical tokens and modifying them by character-level transformations.
edit-based	[38]	black-box	non-targeted	Using natural and synthetic noises to replace the correct words.
	[39]	black-box	non-targeted	Using nine attack modes including visual and phonetic adversaries.

ever, it needs further verification whether *DISTFLIP* distills all knowledge of a white-box attack.

3.2.2. Importance-based method

The importance-based *DeepWordBug* [45] includes two procedures: determining critical tokens, and modifying them slightly. To find important tokens, *DeepWordBug* uses four token scoring functions: *Replace-1 Score*, *Temporal Head Score*, *Temporal Tail Score*, and *Combination Score*. In the second phase, the character-level transformations, such as swap, substitution, deletion, and insertion, are applied to the highest-ranked token to minimize the edit distance of the perturbation. Although *DeepWordBug* successfully generates adversarial texts, most of the introduced perturbations are constricted to misspellings.

3.2.3. Edit-based method

Belinkov et al. [38] utilized the natural and synthetic noise to replace corresponding correct words in a text. They collected natural noises like typos and misspellings from different datasets. Moreover, they crafted synthetic noises through four operations: exchanging characters, randomizing all but the first and last characters of a word, randomizing all characters of a word, and replacing a character with its neighbor on the keyboard. Furthermore, considering the realization in typical application scenarios such as social media, Eger et al. [39] proposed the first large-scale catalog and benchmark of low-level adversarial attack, called *Zéroe*. It encompasses nine attack modes, including visual and phonetic adversaries: inner-shuffle, full-shuffle, intrude, disemvowel, truncate, segment, typo, natural noise, phonetic, and visual.

3.3. Word-level attack

The word-level attack modifies words in a given text. Thus, the modified object is the most significant difference between the word-level attack and the character-level one. The word-level attack often adopts four example generation strategies: gradient-based, optimization-based, importance-based, and edit-based methods, as summarized in Table 3. Optimization-based, importance-based, and edit-based methods are designed for textual adversarial attacks. Overall, word-level methods usually maintain semantic consistency and imperceptibility better than other attacks, but their generated adversarial examples are less varied.

3.3.1. Gradient-based method

Researchers utilized the gradient in various ways. Papernot et al. [34] first evaluated the forward derivative associated with the embedding input of a word sequence through the computational graph unfolding technique and Jacobian Based Saliency Map. Then, they utilized the Fast Gradient Sign Method (FGSM) to craft adversarial perturbations. To solve the problem of mapping generated vectors to nonexistent words, they set up a specific dictionary for replacing the nonexistent word with an appropriate one. However, their method commonly leads to grammatical errors and semantic changes. *TextFool* [65] uses the magnitude of the victim model's cost gradient to determine what, where, and how to insert, replace, or remove words in white-box scenarios. Besides, it utilizes three modification strategies (insertion, modification, and removal) and adopts the natural language watermarking technique to dress up a given text elaborately. However, *TextFool* has massive manual executions. *AdvGen* [119] leverages the gradient itself. It considers the final translation loss of the victim model and the distance between a word and its adversarial one. Besides, it applies a language model to identify possible substitutes for a given word to enhance the semantic-preserving.

Since ensuring the generated vector map to a readable word is critical, Michael et al. [70] demonstrated "*adversarial examples should be meaning-preserving on the source side, but meaning-destroying on the target side*" for non-targeted attacks. They proposed gradient-based word substitution methods with *kNN* and *CharSwap* constraint, respectively. *Seq2Sick* [64] is a projected gradient method combined with group lasso and gradient regularization. The non-overlapping attack requires adversarial examples to share no overlapping words with the original one, while the targeted keyword attack requires adversarial examples to contain all given targeted keywords. Thus, *Seq2Sick* applies a hinge-like loss function optimized at the logit layer and an additional mask function m_t for them, as shown below:

$$L_{non-overlapping} = \sum_{t=1}^{M} \max\left\{-\epsilon, z_t^{(s_t)} - \max_{y \neq s_t}\left\{z_t^{(y)}\right\}\right\} \tag{5}$$

$$L_{keywords} = \sum_{i=1}^{|K|} \min_{t \in [M]}\left\{m_t\left(\max\left\{-\epsilon, \max_{y \neq k_i}\left\{z_t^{(y)}\right\} - z_t^{(k_i)}\right\}\right)\right\} \tag{6}$$

3.3.2. Optimization-based method

In the early stage, Kuleshov et al. [120] was inspired by the concept of *thought vector* and proposed an iterative method, which replaces the original word with the nearest neighbor having the most prominent influence on the objective function in each iteration. Sato et al. [121] proposed *iAdv-Text*, whose optimization process is shown as:

$$\mathscr{J}_{iAdvT}(D, W) = \frac{1}{|D|} \times \arg\min_W \left\{ \sum_{\left(\widehat{X}, \widehat{Y}\right) \in D} \ell\left(\widehat{X}, \widehat{Y}, W\right) + \lambda \sum_{\left(\widehat{X}, \widehat{Y}\right) \in D} \alpha_{iAdvT} \right\} \tag{7}$$

where α_{iAdvT} is the maximization process to find the worst-case weights of the direction vectors. Besides, *iAdv-Text* uses the cosine similarity to maintain the readability and semantic similarity. Gong et al. [122] applied the FGSM and *DeepFool* [124] to find adversarial perturbations in the word embedding space, and used *WMD* to map

Table 3
Summary of Word-level Adversarial Attacks in NLP.

Strategy	Work	White/ Black-box	Targeted/ Non-targeted	Main Ideas
gradient-based	[34]	white-box	both	Using FGSM to craft adversarial vectors, setting up a dictionary to replace nonexistent words.
	[65]	both	non-targeted	Designing three perturbing strategies, using the natural language watermarking technique and the magnitude of cost gradient to craft adversarial texts.
	[119]	white-box	non-targeted	Using the gradient itself of the victim model, using the language model.
	[70]	white-box	non-targeted	Word substitutions with *kNN* and *CharSwap* constraints.
	[64]	white-box	targeted	Non-overlapping and Keywords attack, with Group Lasso Regularization and Gradient Regularization.
optimization-based	[120]	white-box	non-targeted	Introducing the thought vector to an iterative procedure to replace words with the nearest neighbor each time.
	[121]	black-box	non-targeted	Optimization process on embedding space with cosine similarity constraint.
	[122]	black-box	non-targeted	Optimization process on embedding space with *WMD* constraint.
	[123]	white-box	targeted	Using antonym substitution strategy, locating and replacing words with antonyms.
	[40]	black-box	non-targeted	An optimization method based on the GA, randomly selecting the word to be replaced.
	[41]	black-box	non-targeted	Based on the work in [40], allowing a word to be modified multiple times.
	[72]	black-box	non-targeted	The GA based optimization with a multi-objective strategy.
	[71]	black-box	non-targeted	The GA based optimization with word replacement strategy.
	[47]	black-box	non-targeted	Reducing search space by a sememe-based word replacement method, searching for adversarial examples by the PSO in the reduced search space.
	[37]	black-box	non-targeted	Regarding the operations of identifying key words and selecting substitutes as *action*, and using Policy Gradient to learn a *policy*.
	[76]	black-box	non-targeted	Perturbing words' inflectional morphology to craft plausible and semantically similar adversarial texts.
	[77]	black-box	non-targeted	Using randomized mechanisms satisfying *metric-DP* to filter out substitutes with irrelevant meanings.
	[73]	black-box	non-targeted	Leveraging the GA to find an optimal ensemble with the minimum number of model members.
importance-based	[78]	black-box	non-targeted	Determining the word importance by the probability change value when the word is deleted, modifying the input by a removal-addition-replacement strategy.
	[81]	black-box	non-targeted	Synonym substitution method using the word saliency and classification probability.
	[82]	black-box	non-targeted	Using the cross-domain interpretability to learn the word importance.
	[79]	black-box	non-targeted	Identifying important words, and replacing them with the most semantically similar and grammatically correct substitutes.
	[80]	black-box	non-targeted	Identifying important words, and replacing them by considering the information of both the original word and its surrounding context.
	[83]	both	non-targeted	A systematic probabilistic framework, and its two instantiations: *Greedy Attack* and *Gumbel Attack*.
edit-based	[84]	both	non-targeted	Employing the Metropolis–Hastings sampling approach and a language model to craft fluent adversarial texts.
	[85]	black-box	non-targeted	Using a mask-then-infill procedure with three contextualized perturbations to craft adversarial texts.
	[86]	black-box	non-targeted	Detecting word sense disambiguation biases in neural machine translation models.

the adversarial example to the nearest meaningful word vector. Additionally, Song et al. [123] proposed a three-step adversarial example generation method for optical character recognition models. It first finds words and their antonyms in *WordNet* [125] and only remains valid and semantically consistent antonyms satisfying the edit distance threshold. Then, it locates lines containing the above words in the clean image and transforms the target words in these lines to appropriate ones through the L_2-norm distance. Finally, it replaces images of the corresponding lines in the text image with adversarial ones.

Later, researchers employed various evolutionary algorithms in the adversarial text generation procedure. Alzantot et al. [40] utilized GA [74] to select words randomly and find their nearest neighbors. They ranked and substituted the selected word to maximize the target label's probability. Wang et al. [41] improved the work in [40] by allowing the words in a given sentence to be modified multiple times. Maheshwary et al. [71] leveraged a GA-based approach in a hard label black-box setting. Mathai et al. [72] proposed a GA-based optimization method with a multi-objective strategy. However, randomly selecting words for substitution in the above methods is full of uncertainties, making some changes meaningless for the target label. Differently, Zang et al. [47] leveraged the PSO [75] to determine the word to be modified. They further demonstrated that the substitute found based on the word embedding and language model is not always semantically consistent with the replaced word or suitable for the context. In addition, they proposed a sememe-based word substitution method. The adversarial example generation procedure of this method is shown in Fig. 5.

Besides, considering that second-language speakers and many first-language dialect speakers frequently exhibit variability in their production of inflectional morphology, *MORPHEUS* [76] maximally increases the prediction loss by searching for the inflectional form of each noun, verb, or adjective in a given text greedily. Xu et al. [77] presented that optimizing the worst-case loss function over all possible substitutions is prone to weigh unlikely substitutions higher and limit the accuracy gain. Thus, they proposed a Metric Differential Privacy mechanism, which samples k values from a truncated poisson distribution as substitution candidates to ensure nearby words with irrelevant meanings are disregarded. For a given privacy parameter, an irrelevant word could have a similar substitution probability as a relevant word. Hence, this method is with different degrees of semantic preservation.

More generally, considering the inapplicability of massive queries in the real world, Zang et al. [37] introduced RL to learn from the attack history. They regarded two operations (identifying vital words to be substituted, and selecting an appropriate substitute to replace the identified vital word) as the *action*. They then used the Policy Gradient to learn the *policy* under which an adversarial example is crafted by taking a series of *actions*. This method theoretically can be combined with any candidate substitute method. Yuan et al. [73] systematically studied the transferability of adversarial attacks. They leveraged the GA to find an optimal ensemble with the minimum number of model members to generate adversarial texts that strongly transfer to other victim models. Further, they generalized adversarial examples constructed by the

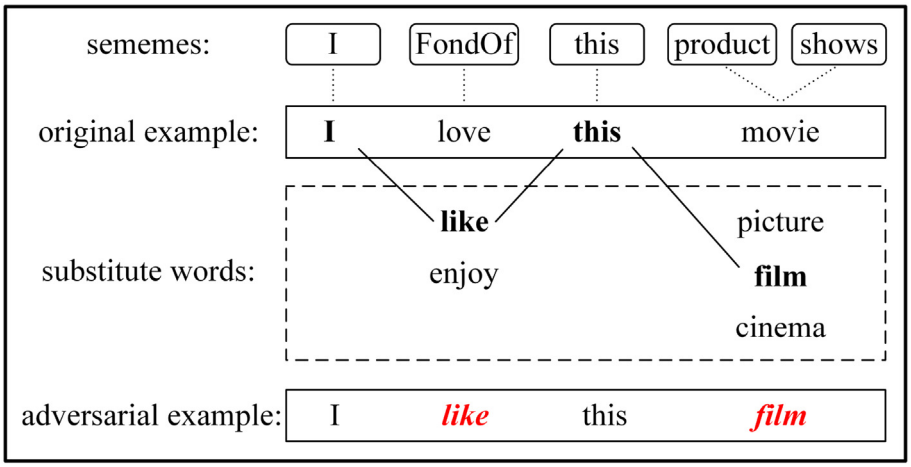

Fig. 5. The procedure of method [47]. It first uses the sememe-based word replacement method to exclude the invalid or low-quality substitutes. Thus, the remaining ones form the reduced search space. Then, it uses the PSO-based search method to efficiently find adversarial examples in the reduced search space.

ensemble method into universal semantics-preserving word replacement rules, which induce adversaries on any text input.

3.3.3. Importance-based method

For generating semantics-preserving texts with minimum modifications, Samanta et al. [78] first ranked words in descending order according to the class probability changes, which were obtained by removing words one by one. Then, they modified the input with a *removal-addition-replacement* strategy. It works best for datasets such as the Internet Movie Database (IMDB), which has sub-categories within each class. Later, Jin et al. [79] and Maheshwary et al. [80] adopted the keyword ranking method in [78]. The difference is, Jin et al. [79] utilized three strategies (synonym extraction, part-of-speech checking, and semantic similarity checking) to replace words with the most semantically similar and grammatically correct substitutes. While Maheshwary et al. [80] further considered the original word and its surrounding context when searching for substitute candidates. Besides, Ren et al. [81] proposed a synonym substitution based *Probabilistic Weighted Word Saliency* (*PWWS*) method, which determines the word replacement order by the word saliency and classification probability. The former reflects the importance of the original word to classification probability; the latter indicates the attack performance of the substitute. This method performs well in effectively reducing substitutes.

For handling the problem of computational complexity and query consumption, *Explain2Attack* [82] employs cross-domain interpretability to gain word importance in black-box scenarios. They first built an interpretable substitute model that imitates the victim model's behavior. Then, they used the interpretability capability to produce word importance scores. It reduces computational complexity and query consumption to a large extent while ensuring the attack success rate.

More generally, Yang et al. [83] proposed a systematic probabilistic framework, in which critical features are identified at first and then perturbed with values chosen from a dictionary. As two instantiations of this framework, *Greedy Attack* crafts single-feature perturbed inputs that achieve a higher success rate, *Gumbel Attack* learns a parametric sampling distribution and requires fewer model evaluations, leading to better efficiency in real-time or large-scale attacks.

3.3.4. Edit-based method

The *Metropolis Hastings Attack* (*MHA*) [84] employs the Metropolis-Hastings sampling approach to replace old words and random words. The only difference between black-box and white-box *MHA* is the pre-selected function used for selecting the most likely word to modify. Compared with previous language generation models using the Metropolis-Hastings sampling approach, the black-box *MHA* 's stationary distribution is equipped with a language model term and an adversarial attacking term, making the adversarial text generation fluent and effective.

Furthermore, Li et al. [85] built their model on a pre-trained masked language model and modified the input in a context-aware manner. They proposed three contextualized perturbations (replace, insert, and merge) and used a mask-then-infill procedure to generate fluent and grammatical adversarial texts with varied lengths.

Emelin et al. [86] detected word sense disambiguation bias in neural machine translation models for model-agnostic adversarial attacks. The word sense disambiguation is a well-known source of translation errors in neural machine translation tasks. They thought that some incorrect disambiguation choices result from models' over-reliance on dataset artifacts found in training data, specifically superficial word co-occurrences. Besides, they minimally perturbed sentences to elicit disambiguation errors to probe the robustness of translation models. It does not require access to gradient information or the score distribution of the decoder.

3.4. Sentence-level attack

The sentence-level attack takes the sentence as the object and includes operations like inserting new sentences, generating its paraphrase, and even changing its structure. Slightly different from the former two kinds of attacks, the sentence-level attack frequently employs five example generation strategies: gradient-based, optimization-based, edit-based, paraphrase-based, and generative model-based methods, as shown in Table 4. Adversarial examples generated by these methods are semantics-preserving and full of diversity, but some of them have reduced readability caused by adding the meaningless token sequence.

3.4.1. Gradient-based method

The universal adversarial perturbation is a particular noise, combined with which any text can fool NLP models with a high probability. Behjati et al.[87] is the first to investigate the universal adversarial perturbations in the NLP field. They used an iterative projected gradient-based approach on embedding space to craft a sequence of words. Then, they applied the generated sequence to any input sequence in the corresponding domain. The *Natural Universal Trigger Search* method [88] is based on an adversarially regularized autoencoder [126]. In order To avoid out-of-

Table 4
Summary of Sentence-level Adversarial Attacks in NLP.

Strategy	Work	White/ Black-box	Targeted/ Non-targeted	Main Ideas
gradient-based	[87]	both	both	Using an iterative projected gradient based approach to find universal perturbations.
	[88]	black-box	Non-targeted	Based on an adversarially regularized autoencoder, and leveraging the projected gradient descent with l_2 regularization.
	[89]	white-box	non-targeted	Using *HotFlip* to generate universal adversarial triggers by an optimization process guided by the gradient.
	[90]	white-box	non-targeted	Extending *HotFlip* by jointly minimizing two losses to generate universal triggers, then using a conditional language model to craft adversarial texts.
optimization-based	[118]	black-box	non-targeted	A combinatorial optimization that maximizes the quantity measuring the degree of violation, and using language models to generate linguistically plausible examples.
edit-based	[58]	black-box	non -targeted	Generating and adding a distractor sentence to the end of a given paragraph.
	[46]	black-box	non-targeted	Distractors are randomized placed to increase the variance of adversarial examples.
	[59]	black-box	non-targeted	Using model extraction to approximate the victim model, using *ADDANY-KBEST* improved from *ADDANY* to generate adversarial examples.
paraphrase-based	[63]	black-box	non-targeted	An encoder-decoder framework that generates adversarial sentences with a target syntax structure.
	[91]	black-box	non-targeted	Generating paraphrases with semantically equivalent rules, and regarding the paraphrase as an adversarial example.
	[42]	black-box	non-targeted	Expanding texts by inserting an adversarial modifier for each constituent determined by linguistic rules.
generative model-based	[94]	black-box	non-targeted	A framework consisting of a GAN and a converter, and using iterative stochastic search and hybrid shrinking search to search for adversarial examples.
	[36]	black-box	non-targeted	A framework consisting of a GAN and an autoencoder, and using the RL to guide the training of the GAN.
	[97]	black-box	non-targeted	Generating adversarial texts through controllable attributes that are known to be irrelevant to the task label.

distribution noise vectors and maintain the naturalness of generated texts, this method leveraged the projected gradient descent with l_2 regularization.

Furthermore, several gradient-guided methods, which are based on *HotFlip* [25], are proposed to generate universal adversarial triggers. For instance, Wallace et al. [89] first specified the trigger length and initialized a sequence trigger, then replaced tokens of the initialized sequence through *HotFlip*, finally added the generated trigger to the beginning or end of the given text. Due to that, a long trigger is more effective and more noticeable than a shorter one; the trigger length in this method is an important criterion. Atanasova et al. [90] focused on ensuring the semantic validity of adversarial texts. They extended *HotFlip* by jointly minimizing a fact-checking model's target class loss and an auxiliary natural language inference model's entailment class loss to generate universal triggers. Then, the generated universal triggers were input to a conditional language model trained using a *GPT-2* model. Their method effectively crafts semantically valid statements containing at least one trigger.

3.4.2. Optimization-based method

Minervini et al. [118] studied the automatic generation issue of adversarial examples that violate a set of given First-Order Logic constraints in the natural language inference task. They regarded this issue as a combinatorial optimization problem. They generated linguistically plausible texts by using language models and maximizing the quantity that measures the violation degree of such constraints. Specifically, they maximized the inconsistent loss J_I to search for the substitution set S (i.e., adversarial examples) using the following language model:

$$\max_S x_I(S) = [p(S;body) - p(S;head)]_+ \quad \text{s.t. } \log p_L(S) \leqslant \tau \tag{8}$$

Here, $p_L(S)$ represents the probability of a sentence in $S = \{X_1 \to s_1, \ldots, X_n \to s_n\}$, τ is a threshold on the perplexity of generated sequences. $p(S;body)$ and $p(S;head)$ are probabilities of the given rule, after replacing X_i with the corresponding sentence S_i, *body* and *head* represent the premise and the conclusion of the natural language inference rules.

3.4.3. Edit-based method

Jia et al. [58] proposed *ADDSENT* and *ADDANY*, both of them generate distractor sentences that confuse models but do not contradict the correct answer or confuse humans. *ADDSENT* generates a sentence looking similar to the question but does not contradict the correct answer, and then adds the generated sentence to the end of the given paragraph. Its variant *ADDONESENT* adds random sentences recognized by human beings. On the contrary, *ADDANY* does not consider the sentence's grammar and adds an arbitrary sequence of English words by querying the victim model many times. Unlike *ADDSENT* and *ADDANY*, which both try to incorporate words in the question into the adversarial sentence, *ADDCOMMON*, a variant of *ADDANY*, only uses common words in the adversarial sentence.

Based on the work in [58], Wang et al. [46] proposed *ADDENT-DIVERSE*, an improvement of *ADDSENT*, to craft adversarial examples with a higher variance where distractors will have randomized placements, leading to the expansion of the fake answer set. To address the antonym-style semantic perturbations used in *ADDSENT*, they added semantic relationship features to enable the model to identify the semantic relationship among question contexts with the help of *WordNet*. Further, Nizar et al. [59] approximated a black-box victim model via the model extraction [127], and then used *ADDANY-KBEST*, a variant of *ADDANY*, to craft adversarial examples.

3.4.4. Paraphrase-based method

Some researchers have generated the paraphrase of a given text as its adversarial example. Ribeiro et al. [91] iteratively generated paraphrases for an input sentence and obtained the victim model prediction until the prediction was changed. They proposed a semantic-equivalent rule-based method to generalize these generated examples into semantically equivalent rules for understanding and fixing the most impactful bugs. When generating the paraphrase, controlled perturbations are incorporated. Iyyer et al. [63] proposed an encoder-decoder based *SCPNs* method. For a

given sentence and a corresponding target syntax structure, *SCPNs* first encodes the sentence by a bidirectional Long Short Term Memory (LSTM) model, and then inputs the interpretation and targeted syntactic tree into the LSTM model for decoding to obtain the targeted paraphrase of the given sentence. In the decoding procedure, the soft attention [92] and copy mechanism [93] are further introduced. Although *SCPNs* uses the target strategy, it does not specify the target output. Besides, adversarial texts generated by *SCPNs* can effectively improve the pre-trained model's robustness to syntactic changes.

Some other researchers generated adversarial examples by expanding the original sentence. For example, *AdvExpander* [42] first uses linguistic rules to determine which constituents (word or phrase) to expand and what types of modifiers to expand with. Then, it expands each component by inserting an adversarial modifier searched from a conditional variational autoencoder based generative model [128] that is pre-trained on the Billion Word Benchmark. This method differs from existing substitution-based methods and introduces rich linguistic variations to adversarial texts.

3.4.5. Generative model-based method

Zhao et al. [94] proposed a GAN-based framework to craft efficient and natural adversarial texts. This framework consists of two main components: a GAN for generating fake data, and a converter for mapping input to its potential representation z'. The two components are trained on original input by minimizing reconstruction errors between original and adversarial ones. The perturbation is performed on the latent dense space by identifying the perturbed example $\hat{z}$ in the neighborhood of z'. Two search approaches (iterative stochastic search, and hybrid shrinking search) were used to identify the proper $\hat{z}$. This method is appropriate for both image and textual data, as it intrinsically eliminates the problem raised by the discrete attribute of textual data. However, due to the requirement of querying the victim model each time to find the $\hat{z}$ that can make the model produce incorrect prediction, this method is quite time-consuming. Furthermore, Wong et al. [36] proposed a GAN-based framework that utilizes RL to guide the training of GAN. Thus, there is no converter but an autoencoder, which judges the semantic similarity between original texts and adversarial ones. However, it is restricted to binary text classifiers.

Differ from the above methods, Wang et al. [97] proposed *CAT-Gen*. Given a text, *CAT-Gen* generates adversarial texts through controllable attributes known to be irrelevant to the task label. As shown in Fig. 6, regarding the product category as a controllable attribute for the sentiment analysis task, *CAT-Gen* first pre-trains an encoder and a decoder to copy input sentence x with the attribute a, and pre-trains an attribute classifier using an auxiliary dataset. Then, given the desired attribute $a' \neq a$, it uses the attribute classifier to train the decoder to enable the model to generate an output with the attribute a'. Finally, it searches through the whole attribute space of $a' \neq a$ and looks for a^* that maximizes the cross-entropy loss between the prediction and ground-truth task-label.

3.5. Multi-level attack

The objects modified in multi-level attacks are two or three kinds of character, word, and sentence. According to the primary strategy used in multi-level attacks, these approaches are divided into the gradient-based, optimization-based, importance-based, edit-based, and generative model-based methods, as shown in Table 5. These methods generated various adversarial examples to a certain extent, but they always have more constraints.

3.5.1. Gradient-based method

HotFlip [25] modifies the character and word in a given text. It performs the atomic-flip operation relying on gradient computation on the one-hot representation. For character-level attacks, the flip operation is represented as:

$$\vec{v}_{ijb} = \left(\ldots;\left(\overrightarrow{0},\ldots(0,\ldots,0,-1,0,\ldots,1,0)_j\ldots,\overrightarrow{0}\right)_i;\ldots\right) \tag{9}$$

The above formula means that the j-th character of i-th word in a text is changed from a to b, which are both characters at a-th and b-th places in the alphabet. The change from the directional derivative along this vector is calculated to find the biggest increase in loss $J(x,y)$. The calculation process is shown below:

$$\max \nabla_x J(x,y)^T \cdot \vec{v}_{ijb} = \max_{ijb} \frac{\partial J^{(b)}}{\partial x_{ij}} - \frac{\partial J^{(a)}}{\partial x_{ij}} \tag{10}$$

where x_{ij} is a one-hot vector that denotes the j-th character of i-th word. For word-level attacks, the *HotFlip* is used further with a few semantics-preserving constraints like cosine similarity. However, with one or two flips under strict constraints, the *HotFlip* only generates a few successful adversarial examples, making it unsuitable for a large-scale attack.

3.5.2. Optimization-based method

For crafting adversarial texts with sentence-level rewriting, Xu et al. [43] first designed a sampling method, called *RewritingSampler*, to efficiently rewrite the original sentence in multiple ways. Then, they allowed for both word-level and sentence-level changes. In order To constrain the semantic similarity and grammatical quality, they employed the word embedding sum and a *GPT-2* model [62].

3.5.3. Importance-based method

TextBugger [24] is used in black-box and white-box scenarios. In white-box scenarios, it is also a gradient-based method, as it first uses the Jacobian matrix J to calculate the importance of each word, as below:

$$C_{x_i} = J_{F(i,y)} = \frac{\partial F_y(x)}{\partial x_i} \tag{11}$$

Here, $F_y(\cdot)$ represents the confidence score of class y, C_{x_i} is the importance score of the i-th word in input x. Then, *TextBugger* uses five editing strategies (insertion, deletion, swap, substitution with visually similar words, and substitution with semantically similar words) to generate character-level and word-level adversarial texts. In black-box scenarios, *TextBugger* split the document into sequences at first. Then, it queries the victim model to filter out sentences with predictions different from the original labels, sorts these sequences in reverse order according to their confidence, and calculates the word importance score through the following equation by deleting:

$$\begin{aligned} C_{x_i} &= F_y(x_1,\ldots,x_{i-1},x_i,x_{i+1},\ldots,x_n) \\ &\quad - F_y(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n) \end{aligned} \tag{12}$$

Finally, the same editing operation as the white-box attack is used to modify texts. In *TextBugger*, only a few editing operations mislead the classifier.

Since previous work exclusively focused on semantic tasks, Zheng et al. [44] is the first to explore adversarial attacks in syntactic tasks. They focused on the dependency parsing model and constructed syntactic adversarial examples at both sentence and phrase levels. They followed a two-step procedure: 1) choosing weak spots (or positions) to change; 2) modifying them to maximize the victim model's errors in both black-box and white-box scenarios.

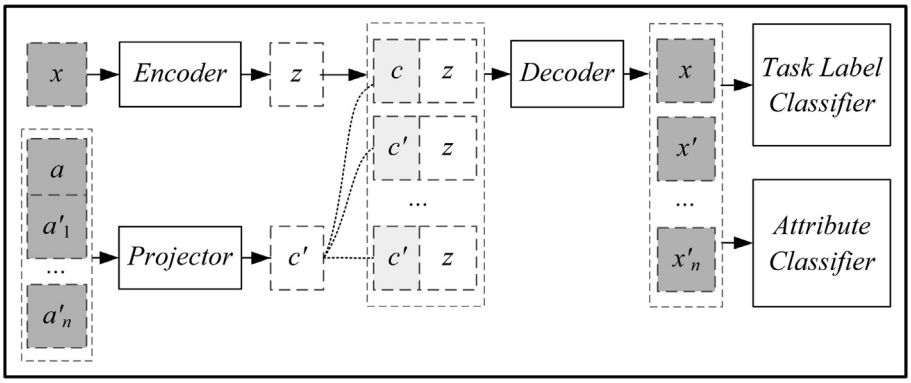

Fig. 6. Overview of *CAT-Gen* training process in [97]. The encoder, decoder, projector, and attribute classifier are pre-trained in advance.

Table 5
Summary of Multi-level Adversarial Attacks in NLP.

Strategy	Work	White/ Black-box	Targeted/Non-targeted	Mian Ideas
gradient-based	[25]	white-box	non-targeted	Modifying characters and words, and performing an atomic-flip operation by gradient computation on vectors.
optimization-based	[43]	black-box	non-targeted	For word-level and sentence-level changes, and using *RewritingSampler* sampling method to rewrite the given sentence.
importance-based	[24]	both	non-targeted	Generating character-level and word-level adversarial texts according to the importance in black-box and white-box scenarios.
	[44]	both	both	Generating phrase-level and sentence-level adversarial texts according to the importance and greedy search strategy in black-box and white-box scenarios.
edit-based	[98]	black-box	non-targeted	Modifying sentences and words by the *Should-Not-Change* and *Should-Change* adversarial strategies.
	[99]	both	non-targeted	Modifying sentences and words by the sentence attention distribution and word importance.
generative model-based	[100]	black-box	non-targeted	Using a hybrid *Adversarial Examples Generator* to introduce both word and character-level perturbations.

3.5.4. Edit-based method

Niu et al. [98] paid attention to both *Should-Not-Change* and *Should-Change* adversarial strategies. The *Should-Not-Change* strategy includes four edits: transposing neighboring tokens randomly, removing stopwords randomly, replacing words with their paraphrases, and using grammar errors like changing a verb to the wrong tense. The *Should-Change* strategy contains two methods: negating the root verb of the source input and changing verbs, adjectives, or adverbs to their antonyms.

Blohm et al. [99] implemented four adversarial text generation methods. For black-box attacks, they performed the word-level attack by manually replacing original words with substitutes in pre-trained *Glove* embedding, and used the *ADDANY* [58] as sentence-level attacks. For white-box attacks, they leveraged the models' sentence-level attention distribution to find the plot sentence, which has the greatest attention. The k words receiving the most attention in the plot sentence were exchanged by randomly chosen words in word-level attacks. Finally, the whole plot sentence was removed in sentence-level attacks.

3.5.5. Generative model-based method

Vijayaraghavan et al. [100] developed a hybrid encoder-decoder model, called the *Adversarial Examples Generator*. It consists of two components: an encoder and a decoder. The encoder, which is a slight variant of Chen et al. [129], maps the input sequence to a representation using word and character-level information. The decoder has two-level Gate Recurrent Units (GRU): word-GRU and character-GRU. For training the model, they used the self-critical approach of Rennie et al. [130] as their policy gradient training algorithm. Compared with Wong et al. [36], the most outstanding advantage of this method is introducing both word and character-level perturbations.

3.6. Comparison and analysis of attacks

To give readers a more intuitive understanding of these attack methods, this subsection firstly shows the adversarial examples generated by several representative attacks, and then compares the attack performance and query time consumption of these methods. Specifically, the BERT model [131] pre-trained on Stanford Sentiment Treebank (SST) dataset is the victim model, which performs the sentiment analysis task and outputs probabilities for label *Positive* and *Negative*. We utilize 1 character-level attack (*DeepWordBug* [45]), 3 word-level attacks (*Probabilistic Weighted Word Saliency* (*PWWS*) [81], *TextFooler* [79], and BERT-based attack [132]), 2 sentence-level attacks (*Synthetically Controlled Paraphrase Networks* (*SCPNs* [63]), and GAN-based attack [94]), and 1 multi-level attack (*TextBugger* [24]) to attack the chosen victim model. Fig. 7 and Table 6 show generated adversarial examples and attack performances of these methods, respectively.

From the perspective of example quality, character-level attack methods maintain the semantics of original texts well. However, they are easily detected by human eyes or spelling check tools. For example, in the *TextBugger*, words "*crahm*", "*aovids*", "*obv1ous*", and "*hmour*" in the adversarial example are easily observed. Nevertheless, they are less likely to affect the human eye's judgment of the emotions of the whole text. In contrast, word-level attacks compensate for the vulnerability of adversarial examples to detection but affect the semantics of the text to some extent. For instance, in the text "*division of the spell of satin rouge is that it void the obvious with body and weightlessness.*" generated by *PWWS*, the

Original Example:	**Label**
1) Part of the charm of Satin Rouge is that it avoids the obvious with humour and lightness .	*Positive*(99.58%)
2) Just the labour involved in creating the layered richness of the imagery in this chiaroscuro of madness and light is astonishing .	*Positive*(99.27%)
***DeepWordBug*:**	***Prediction***
1) part of the charm of satin rouge is that it avoids the obvious with humour and lightness .	*Negative*(61.23%)
2) just the labour involved in creating the **layere** richness of the imagery in this chiaroscuro of madness and light is astonishing .	*Negative*(85.33%)
***PWWS*:**	***Prediction***
1) **division** of the **spell** of satin rouge is that it **void** the obvious with **body** and **weightlessness**.	*Negative*(71.00%)
2) just the labour involved in creating the layered **cornucopia** of the imagery in this chiaroscuro of **foolishness** and light is **astound** .	*Negative*(50.80%)
***TextFooler*:**	***Prediction***
1) **office** of the **spell** of satin **blusher** is that it **forfend** the obvious with **body** and **lightsomeness** .	*Negative*(82.12%)
2) just the labour involved in creating the **superimposed cornucopia** of the **imaging** in this chiaroscuro of madness and **lighter** is **astound** .	*Negative*(54.21%)
***BERT-Based Attack*:**	***Prediction***
1) *Failed to attack !*	
2) just the labour involved in creating the layered richness of the imagery in this chiaroscuro of madness and light is **enigma** .	*Negative*(73.52%)
***SCPNs*:**	***Prediction***
1) *Failed to attack !*	
2) *in this sense of the richness of the great richness of the image of the image of madness and light , only the work involved in of by in from within of by in from within of by*	*Negative*(57.22%)
***GAN-Based Attack*:**	***Prediction***
1) *Failed to attack !*	
2) *the the of in the the of of the in the film of a*	*Negative*(57.22%)
***TextBugger*:**	***Prediction***
1) Part of the **crahm** of Satin rouge is that it **aovids** the **obv1ous** with **hmour** and **lifhtness** .	*Negative*(73.49%)
2) Just the labour involved in creating the layered richness of the imagery in this chiaroscuro of madness and light is **astnishing** .	*Negative*(92.08%)

Fig. 7. Adversarial examples generated by several attacks. The original examples are two randomly selected from the SST dataset, and both of them are correctly classified as *Positive* by the pre-trained BERT model.

Table 6
Comparison of Adversarial Attack Performance.

Method	Attack Success Rate	Average Model Queries (times)	Average Running Time (seconds)
DeepWordBug	59.46%	23.626	**0.000439**
PWWS	75.74%	117.82	0.002190
TextFooler	74.86%	61.68	0.053360
BERT-based attack	88.44%	61.94	0.036131
SCPNs	75.66%	11.75	2.366100
GAN-based attack	42.06%	**2.42**	0.009495
TextBugger	**90.54%**	48.79	0.001722

meaning of "*body*" and "*weightlessness*" are very different from the meaning of replaced words "*humour*" and "*lightness*". These examples crafted by word-level methods also lacked diversity. Thus, sentence-level attacks enhance the diversity of generated examples. However, it is clear to see that these adversarial examples crafted by sentence-level *SCPNs* and GAN-based methods are very different from the original ones in both semantics and readability. Therefore, generating semantically consistent and imperceptible adversarial examples at the sentence level might be one of the future research directions.

For comparing the attack performance of the above methods, we randomly selected 5000 examples from the SST dataset to generate corresponding adversarial texts and attack the selected victim model using the above methods. Table 6 shows the result. In terms of attack success rate, *TextBugger* is the highest, and its execution time is also relatively low. The reason might be that *TextBugger* uses the Jacobian matrix to calculate the importance of each word at once. In comparison, the average model queries of sentence-level methods (*SCPNs* and GAN-based method) are

the lowest, but their attack success rates are not satisfactory. As mentioned above, the differences between the adversarial examples generated by sentence-level methods and the original ones are relatively huge, so researchers should focus on maintaining the semantic consistency and imperceptibility of texts for sentence-level methods. In addition, the model query accounts for word-level attacks (*PWWS*, *TextFooler*, and BERT-based attack) are comparatively numerous. Therefore, it is worthwhile for researchers to investigate how to reduce the number of queries about such methods.

4. Textual adversarial attack application

To explore potential adversarial attack threats faced by existing NLP intelligent systems and further provide a basis for developing efficient defense strategies for these NLP models, several researchers applied various textual adversarial attacks to extensive NLP models. While these diverse NLP tasks, such as classification, neural machine translation, text entailment, and dialogue generation, are extremely different and generally implemented on distinct datasets. Thus, the adversarial attack methods applied to these different NLP tasks have their own characteristics. Therefore, this section reviews current works on textual adversarial attacks from the application perspective and summarizes them in Table 7. In detail, Section 4.1 presents popular benchmark datasets in the NLP field, and Section 4.2 elaborates on various applications of textual adversarial attacks.

4.1. Benchmark dataset

Since there are massive benchmark datasets for different NLP tasks, this section gives a brief introduction of the applicable task, dataset size, characteristic, and source to primary datasets in Table 7.

AG's News:[2] used for text classification. It consists of over 1 million news collected from more than 2,000 news sources by an academic news search engine called *Cometomyhead*. In total, it includes 120,000 training examples and 7,600 test examples, which come from four categories of the same scale: World, Sport, Business, and Technology, and each category has 30,000 training examples and 1,900 test examples. The provided *DB* version and *XML* version are available for download for non-commercial use.

DBPedia Ontology:[3] used for text classification. It is a dataset with structured content from the information created in various Wikimedia projects. This dataset contains 560,000 training examples and 70,000 testing examples of 14 high-level classes, such as Company, Building, and Film. It has more than 685 classes represented by a subsumption hierarchy structure and is described by 2,795 different attributes.

Yahoo! Answers:[4] used for text classification. It contains 4 million question-answer pairs, and it can be used in question-answer systems. Furthermore, it includes ten categories of classification data obtained from Yahoo! Answers Comprehensive Questions and Answers 1.0, and each class contains 140,000 training examples and 5,000 test examples.

Stanford Sentiment Treebank (SST):[5] used for sentiment analysis. It includes 239,232 sentences and phrases, whose syntax changes greatly. Compared with most other datasets that ignore word order, SST establishes a complete representation based on the sentence structure. Furthermore, it can judge the mood according to the phrases of words. However, the average example length of SST is only about 17 words.

Internet Movie Database (IMDB):[6] used for sentiment analysis. It is crawled from the Internet, including 50,000 highly polarized movie reviews with tags (positive or negative sentient) and URLs from which the reviews came. Among them, 25,000 examples are used for training and 25,000 for testing. The average length of reviews is nearly 234 words, and the size of this dataset is bigger than most similar datasets. Besides, IMDB also contains additional unlabeled data, original texts, and processed data.

Rotten Tomatoes Movie Review (MR):[7] used for sentiment analysis. It is a labeled dataset that concerns sentiment polarity, subjective rating, and sentences with subjectivity status or polarity. It contains 5,331 positive examples and 5,331 negative examples, and the average example length is 32 words. Since MR is labeled by manual work, its size is smaller than others, with a maximum of dozens of MB.

Amazon Review:[8] used for sentiment analysis. It has nearly 35 million reviews that cross from June 1995 to March 2013, including product and user information, ratings, and plain text reviews. It is collected by over 6 million users in more than 2 million products and categorized into 33 classes with a size ranging from KB to GB.

Multi-Perspective Question Answering (MPQA):[9] used for sentiment analysis. It is collected from various news sources and annotated for opinions or other private states. It contains 10,606 examples, and each example is labeled with objective or subjective sentiment. Three different versions are available to people through the *MITRE* Corporation. The higher the version is, the richer the contents are.

Stanford Question Answering Dataset (SQuAD):[10] used for machine reading comprehension. It contains 107,785 manual-generated reading comprehension questions about more than 500 Wikipedia articles. Each question involves a paragraph of an article, and the corresponding answer is in that paragraph. Compared with SQuAD 1.1, SQuAD 2.0 contains 100,000 questions in SQuAD 1.1 and more than 50,000 malicious seemingly answerable but unanswerable questions written by crowd workers. Thus, SQuAD 2.0 requires machine reading comprehension models to answer questions when possible and determine when paragraphs do not support answers and avoid answers.

MovieQA:[11] used for machine reading comprehension. It aims to evaluate the model's automatic story comprehension ability from both video and text perspectives. This dataset contains 14,944 multiple-choice questions for 408 movies collected by human annotators. Its questions, which vary from simple "who" or "when" to more complex "why" or "how", can be answered by a variety of information sources, including film editing, plot, and subtitles. Each question has five reasonable answers, and only one is correct.

Stanford Natural Language Inference Corpus (SNLI):[12] used for text entailment. It concludes with 570,000 human-written English sentence pairs with a manual label of entailment, contradiction, and neutral. There are 550,152 training examples, 10,000 verification examples, and 10,000 test examples.

Visual Question Answering Dataset (VQA):[13] used for visual question answering. It is the most widely used dataset for visual question answering tasks. Its images are divided into two parts:

[2] https://www.kaggle.com/amanannandrai/ag-news-classification-dataset.
[3] https://dbpedia.org/ontology/.
[4] https://sourceforge.net/projects/yahoodataset/.
[5] https://nlp.stanford.edu/sentiment/code.html.
[6] http://ai.stanford.edu/amaas/data/sentiment/.
[7] http://www.cs.cornell.edu/people/pabo/movie-review-data/.
[8] http://snap.stanford.edu/data/web-Amazon.html.
[9] http://mpqa.cs.pitt.edu/.
[10] https://rajpurkar.github.io/SQuAD-explorer/.
[11] http://movieqa.cs.toronto.edu/home/.
[12] https://nlp.stanford.edu/projects/snli/.
[13] https://visualqa.org/download.html.

Table 7
Summary of Adversarial Attack Applications and Benchmark Datasets in NLP.

Application		Benchmark Dataset	Representative Work
Clasisfication	Text Classification	AG's news	[25,45,81,79,87,83,71,43]
		DBpedia Ontology	[121,65,45]
		Yahoo! Answers	[81,45,83,71]
		Sogou News	[45]
		RCV1	[121]
		Surname Classification Dataset	[133]
	Sentiment Analysis	Yelp Review	[45,120,79,71,43,82]
		SST	[63,47,72,134,87]
		IMDB	[45,121,24,40,78,122,133,81,47,84,72,79,83,71,42,43,82,80]
		MR	[121,24,65,91,79,71,43,82,80]
		Amazon Review	[45,121,82]
		MPQA	[65]
		Elec	[121]
		Arabic Tweets Sentiment	[133]
	Toxic Comment Detection	Toxic Comments dataset (WTC)	[109,60,24,39]
		GOSSIPCOP, PHEME	[135]
	Spam Detection	Enron Spam Email	[45]
		TREC 2007 Public Spam Corpus	[120]
	Relation Extraction	NYT Relation, UW Relation	[136]
		ACE04, CoNLL04 EC, Dutch Real Estate Classifieds, Adverse Drug Events	[137]
	Gender Identification	Twitter Gender	[78]
	Grammar Error Detection	FCE-public	[121]
Neural Machine Translation		TED Talks Parallel Corpus [138]	[35,38]
		WMT corpus	[119,64,86,76,70]
		NIST	[119,86]
Machine Reading Comprehension		SQuAD	[58,59,76,89]
		MovieQA Multiple Choice	[99]
Text Entailment		SNLI	[39,40,42,43,47,71,79,80,84,94,104,118]
		SciTail	[104]
		MultiNLI	[118,134,79,71,43]
		SICK	[63]
		FEVER	[90]
POS Tagging		Penn Treebank WSJ corpus	[44,139,140]
		Universal Dependencies	[139,39]
Text Summarization		DUC2003, DUC2004, Gigaword	[64]
Dialogue Generation		Ubuntu Dialogue Corpus, CoCoA	[98]
Cross-modal	Image Captioning	MSCOCO	[141]
		Visual Genome	[142]
	Optical Character Recognition	Hillary Clinton's emails Corpus	[123]
		Chinese text image dataset	[143]
	Scene Text Recognition	Street View Text, ICDAR 2013, IIIT5	[144]
	Visual Question Answering	VQA dataset	[142,145]
	Visual Semantic Embedding	MSCOCO	[146]
	general V + L task	COCO, Visual Genome, Conceptual Captions, SBU Captions	[147]
	Speech Tecognition	Mozilla Common Voice	[21]
Other Tasks	Interactive Dialogue	–	[148]
	Dependency Parsing	Penn Treebank WSJ corpus	[44]

Table 8
Summary of Applications in Classification Tasks.

Subtask	Work	Benchmark Dataset	Victim Model
Text Clasisfication	[25]	AG's News	CharCNN-LSTM [149]
	[87]	AG's News	LSTM [150], BiLSTM [151], mean-LSTM
	[43]	AG's News	BERT [131]
	[65]	DBpedia Ontology	CharCNN [152], WordCNN [153]
	[45]	AG's News, Sogou News, DBpedia Ontology, Yahoo! Answers	CharCNN, WordLSTM [154]
	[79]	AG's News	WordCNN, WordLSTM, BERT
	[83]	AG's News, Yahoo! Answers	WordLSTM, CharCNN
	[71]	AG's News, Yahoo! Answers	WordCNN, WordLSTM, BERT
Sentiment Analysis	[65]	MR, MPQA	CharCNN, WordCNN
	[78]	IMDB	CNN
	[40]	IMDB	LSTM
	[45]	Amazon Review, Yelp Review	CharCNN, WordLSTM
	[121]	IMDB, Elec, MR, Amazon Review	LSTM based model [155]
	[63]	SST	BiLSTM
	[87]	SST	LSTM, BiLSTM, mean-LSTM
	[84]	IMDB	BiLSTM
	[43]	MR, Yelp Reviews, IMDB	BERT
	[82]	IMDB, MR, Amazon Review, Yelp Review	WordCNN, WordLSTM, BERT
	[47]	IMDB, SST	BiLSTM [156], BERT
	[72]	IMDB, SST	CharLSTM, WordLSTM, ELMo-LSTM
	[79]	MR, IMDB, Yelp Review	WordCNN, WordLSTM, BERT
	[71]	MR, IMDB, Yelp Review	WordCNN, WordLSTM, BERT
	[80]	IMDB, MR	WordLSTM, BERT
Toxicity Comment Detection	[39]	WTC	RoBERTa [157]
	[60]	WTC	Google Perspective API
	[135]	GOSSIPCOP, PHEME	TEXTCNN [153], dEFEND [158]
Spam Detection	[120]	Trec07p	Naive Bayes, CharLSTM, WordCNN
	[45]	Enron Spam Email	WordLSTM, CharCNN
Relation Extraction	[136]	NYT Relation, UW Relation	PCNN [159], BiGRU [160]
	[137]	ACE04, CoNLL04 EC, Dutch Real Estate Classifieds, Adverse Drug Events	BiLSTM [161]
Gender Identification	[78]	Twitter Gender	CNN
Grammatical Error Detection	[121]	FCE-public	BiLSTM [162]

204,721 images that come from the Microsoft COCO dataset (MSCOCO)[14] based on real scenes, and 50,000 pictures consisting of human and animal models in abstract scenes. Humans generate these questions and answers. In particular, the true or false questions account for about 40%, and each picture generally corresponds to several question-answer pairs. At present, there are two versions. In VQA 1.0, the questions are more about the image's simple position, quantity, and attribute relationships. In VQA 2.0, besides the simple attribute information in the picture, the questions are more fused with some concept sense. Thus, more studies tend to use VQA 2.0.

4.2. Application in NLP

Differ from Section 3 focusing on adversarial attack approaches, this section concentrates on the application scenario and impact of adversarial attacks in the NLP field. According to different text-processing tasks, this section classifies adversarial attack applications into eight categories: classification, neural machine translation, machine reading comprehension, text summarization, text entailment, part-of-speech tagging, dialogue generation, and cross-modal tasks. Note that most attacks were simultaneously applied in multiple different tasks, indicating the transferability of these methods across datasets and DNN models.

4.2.1. Classification

The Classification task is one of the most general scenarios in the NLP field. It can be further divided into seven sub-categories: text classification, sentiment analysis, gender identification, grammatical error detection, toxic comment detection, spam detection, and relation extraction, as shown in Table 8.

Text classification aims to categorize the given texts into several classes, such as class "Business" and "Sport" in AG's News dataset. Sentiment analysis classifies sentiments into two or three classes, for example, in a three-group scheme: neutral, positive, and negative. Furthermore, gender identification, grammatical error detection, toxic comment detection, and spam detection can be framed as binary classification problems. In comparison, the relation extraction extracts the corresponding relation of the entity pair in a sentence. Thus, relation extraction can be treated as a multiple classification issue judging the relationship of the entity pair.

From the perspective of the victim model, the LSTM-based models were attacked by several methods [120,45,63,47,84,72,79,87]. Among them, the "Sememe + PSO" attack [47] experimented on the bidirectional LSTM (BiLSTM) model [156] with IMDB and SST dataset. Compared with the "Embedding/Language Model + Genetic" [40] and "Synonym + Greedy" [81] approach, it achieved the highest attack success rate on both datasets, particularly, it attacked BiLSTM/ BERT on IMDB with a notably 100.00%/ 98.70% success rate.

The Convolutional Neural Network (CNN) based models were the target model of several attacks [136,78,65,120,45,25,79,83]. For instance, *HotFlip* [25] was evaluated on the character-level CNN-LSTM (CharCNN-LSTM) model [149] with AG's News in white-box scenarios, and changed an average of 4.18% of the characters to fool the classifier at confidence 0.5. Liang et al. [65] focused on the character-level (CharCNN) [152] and word-level

[14] https://cocodataset.org/#download.

CNN (WordCNN) [153] model, and evaluated their method on DBpedia Ontology, MR and MPQA dataset. For example, by just inserting one word, their method could cause a text that describes a company to be misclassified as the class "Building" with the confidence of 88.6%.

Recently, pre-trained models, such as BERT [131] and ELMo [163], were attacked by adversarial attacks in [43,82,72,39,79,71]. These attacks were essentially implemented with AG's News, MR, Yelp Reviews, and IMDB datasets. For instance, *RewritingSampler* [43] changes the sentence from a passive voice to an active one by replacing four words. Although none of the word substitutes has a similar meaning, the entire semantics of the sentence does not change. Even, the Google Perspective API[15] was attacked by the *DISTFLIP* [60] with the toxic comment dataset[16] in black-box scenarios. 42% of API-predicted labels corresponding to the generated examples were flipped by this attack, while humans maintain high accuracy in predicting the target label.

4.2.2. Neural machine translation

Existing Neural Machine Translation systems are used to translate one natural language into another. Given an adversarial text, the translation obtained from the system is inconsistent with the semantics understood by human beings. Some related works are shown in Table 9.

The adversarial attacks in [38,35] attacked the character-level neural machine translation models on the TED Talks Parallel Corpus [138]. The difference between them is that the former just implemented black-box attacks, while the latter proposed both black-box and white-box attacks. Further, in [35], the average number of character changes and queries in the black-box setting are respectively 3.6 and 4.3 times those in the white-box setting.

To attack the Transformer model [166], Cheng et al. [119] used the gradient-based method and achieved an improvement of 2.8 and 1.6 Bilingual Evaluation Understudy (BLEU) points on the NIST dataset and WMT Corpus, respectively. Besides, Emelin et al. [86] elicited word sense disambiguation biases. They demonstrated that disambiguation robustness varies substantially between domains, and different models trained on the same data are vulnerable to different attacks. Tan et al. [76] perturbed the inflectional morphology of words in given texts. All these attack methods have reduced the performance of DNN models to a large extent.

4.2.3. Machine reading comprehension

In the Machine Reading Comprehension task, the model needs to answer questions based on a given text, and these questions vary from simple "who" or "when" to more complex "why" or "how", which can be answered by a variety of information sources. Further, the machine reading comprehension includes the cloze test, multiple-choice, and free answering tasks. Some adversarial attacks in these tasks are shown in Fig. 10.

Most approaches [58,89,59,76] are evaluated on the SQuAD dataset. Among them, Jia et al. [58] and Wallace et al. [89] attacked the Bi-Directional Attention Flow network (BiDAF) [169], and achieved a nearly 50% attack success rate. In particular, Wallace et al. [89] manually picked the target answers "*to kill american people*", "*donald trump*", "*January 2014*", and "*new york*" for why, who, when, and where questions, respectively. The transferability of these triggers was verified by attacking three black-box models with different embeddings/ tokenizations and architectures, as the attack success rate was about 10% - 30%. Furthermore, Nizar et al. [59] targeted the BERT-based model, and reduced the F1 score on the attacked model by 11 points in comparison to *ADDSENT* [58].

Table 9
Summary of Applications in Neural Machine Translation Tasks.

Work	Benchmark Dataset	Victim Model
[38]	TED Talks Parallel Corpus	Fully character-level model [164], Nematus [165], attentional seq2seq model
[35]	TED Talks Parallel Corpus	CharCNN-LSTM
[119]	NIST, WMT Corpus	Transformer [166]
[86]	NIST, WMT Corpus	Transformer, LSTM-based model [167], ConvS2S [168]
[76]	WMT Corpus	ConvS2S, Transformer
[64]	WMT Corpus	WordLSTM encoder, word-based attention decoder
[70]	WMT Corpus	Transformer, LSTM-based model, ConvS2S

Table 10
Summary of Applications in Machine Reading Comprehension Tasks.

Work	Benchmark Dataset	Victim Model
[58]	SQuAD	BiDAF [169], Match-LSTM [170]
[89]	SQuAD	QANet [171], BiDAF, BiDAF with CharCNN
[59]	SQuAD	BERT based models
[76]	SQuAD	BiDAF, ELMo-BiDAF [163], SpanBERT
[99]	MovieQA Multiple	CNN based, RNN-LSTM based models

Table 11
Summary of Applications in Text Entailment Tasks.

Work	Benchmark Dataset	Victim Model
[63]	SICK	BiLSTM
[90]	FEVER	RoBERTa
[118]	SNLI, MultiSNLI	DAM, ESIM, cBiLSTM
[40]	SNLI	Model with ReLU layers
[84]	SNLI	BiDAF
[43]	SNLI,MultiNLI	BERT
[39]	SNLI	RoBERTa
[47]	SNLI	BiLSTM, BERT
[134]	MultiNLI	BERT, RoBERTa, XLM [172], XLNet [173]
[79]	SNLI, MultiNLI	BiLSTM, ESIM [174], BERT
[71]	SNLI, MultiNLI	BiLSTM, ESIM, BERT
[80]	SNLI	WordLSTM, BERT

4.2.4. Text entailment

In the Text Entailment task, which is also called Natural Language Inference, the machine needs to judge the relationship between a premise text and a hypothesis one. Generally, the relationship can be divided into three categories: entailment, contradiction, and neutral. Some applications are summarized in Table 11.

Intuitively, just a few works used the SICK and FEVER datasets to evaluate the performance of adversarial texts. Among them, the SICK is a relatively small dataset consisting of 10,000 simple sentence pairs, and the FEVER includes 185,445 claims manually verified against the introductory sections of Wikipedia pages and classified as *SUPPORTED*, REFUTED*REFUTED*, or *NOTENOUGHINFO*. Typically, compared with a neural back-translation baseline, *SCPNs* [63] generated adversarial texts with more dramatical variations in syntactic structures on SICK. Most adversarial attacks [118,40,84,43,39,47,134,79,71,42] applied to text entailment tasks are verified on SNLI and MultiSNLI datasets. For instance, Alzantot et al. [40] utilized the GA to craft adversarial examples that main-

[15] https://www.perspectiveapi.com/.
[16] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data.

tain semantics and syntax, and achieved an attack success rate of 70% with a modified rate of 23%. To improve the validity and fluency of adversarial text, Zhang et al. [84] improved the work in [40] by employing the Metropolis-Hastings sampling, and it reduced the perplexity (*PPL*) by nearly 500 points.

From the perspective of the victim model, more and more attacks [43,39,47,134,79,90] were evaluated on the pre-trained language model. Compared with the effective and efficient *TextFooler* [79], *RewritingSampler* [43] showed its outperformance in the diversity, semantic similarity and grammatical quality when attacking the BERT. The lightweight *Mischief* [134] was evaluated on four transformer-based models: BERT [131], RoBERTa [157], XLM [172], and XLNet [173], and significantly reduced the performance by up to 20% of these models. Furthermore, the method in [90] ensured the semantic validity of adversarial texts crafted by universal triggers.

4.2.5. Part-of-speech tagging

The part-of-speech (*POS*) tagging is the process of marking up words in a text as corresponding to a particular part of speech, based on both its definition and context, such as "*noun*" and "*verb*". In particular, Yasunaga et al. [139] added perturbations to the input word or character embedding, and conducted a series of POS tagging experiments on Penn Treebank WSJ corpus [175] (English) and Universal Dependencies [176] (27 languages) dataset. Further, Eger et al. [39] proposed *Zéroe* containing nine attack modes, and evaluated it with the RoBERTa [157] on the Universal Dependencies dataset. The results indicated that the intrude attack, which randomly inserts unobtrusive symbols (like $%&.- and whitespace) in a given text, is among the most severe attacks for POS tagging since it decreased the victim model accuracy to around 16%. Besides, Han et al. [140] used a sequence-to-sequence model with feedback from multiple reference models of the same task to generate adversarial sentences with different lengths and structures.

4.2.6. Text summarization

The Text Summarization task summarizes the main content or meaning of a given document or paragraph with a succinct expression. Because the average length of given texts varies greatly, it is challenging to implement an adversarial attack on this task. For example, *Seq2Sick* [64] was verified on three datasets (the DUC2003, DUC2004, and Gigaword). The DUC2003 and DUC2004 datasets are extensively employed in documentation summarization. Changing only 2 or 3 words on average by *Seq2Sick* leads to entirely different outputs for more than 80% of sentences.

4.2.7. Dialogue generation

The Dialogue Generation is a kind of text generation task, which automatically generates a response according to a given post. Besides, the dialogue generation model is a fundamental component of real-world dialogue systems. Niu et al. [98] verified their approach with the Ubuntu Dialogue Corpus [177] and the Collaborative Communicating Agents dataset (CoCoA) [178]. For the Ubuntu Dialogue Corpus, which contains 1 million 2-person, multi-turn dialogues extracted from Ubuntu chat logs used to provide and receive technical support, they focused on the Variational Hierarchical EncoderDecoder model [179] and the RL model [180]. In contrast, for the CoCoA dialogue dataset, which involves two agents that are asymmetrically primed with a private Knowledge Base (KB) and engage in a natural language conversation to find out the unique entry shared by the two KBs, they paid attention to attack the Dynamic Knowledge Graph Network. Further, adversarial training with generated adversarial examples makes all models more robust to adversarial attacks and improves their performances when evaluated on the original test set.

4.2.8. Cross-modal task

In addition to the tasks dealing with the single modality input, some NLP-related cross-modal tasks are faced with the adversarial attack threat. These cross-modal tasks can be categorized as two main types: text-and-vision, and text-and-audio, as shown in Table 12.

Text-and-Vision. The Image Captioning model takes an image as an input and generates a textual caption, which describes the visual content of the input image. For attacking the CNN-RNN-based image captioning model, Chen et al. [141] proposed *Show-and-Fool*, which includes a targeted caption method that makes model outputs match the target caption, and a targeted keyword method that makes model outputs contain specific keywords. Experimented on the MSCOCO dataset [181], the former achieved a 95.8% attack success rate, and the latter achieved an even higher success rate, especially at least 96% for the 3-keyword case and at least 97% for 1-keyword and 2-keyword cases. Following, Xu et al. [142] proposed an iterative optimization method, and investigated adversarial attacks on the *DenseCap* network [182] with Visual Genome dataset [183]. Its objective function maximizes the probability of the target answer and unweights the preference of adversarial examples with a smaller distance to the original image when this distance is below a threshold. Although it is challenging to train an RNN-based caption generation model to generate the exact matching captions and the *DenseCap* network involves randomness, this method reached beyond 97% success rate.

The Optical Character Recognition model takes an image as an input and outputs the recognized text. These tasks approximately include two types: character-based and end-to-end. The former is a traditional approach for recognizing text in "block of text" images; the latter is a segmentation-free technique that recognizes the entire sequence of characters in a variable-sized "block of text" image. Regarding Tesseract[17] as the victim model, Song et al. [123] successfully caused over 90% of the words in their list to be misrecognized in the character-based task, and flipped the meaning of a relatively long document by changing only 1–2 words in the end-to-end task. Additionally, Chen et al. [143] proposed *WATERMARK*, which produces natural distortion in the disguise of watermarks. In white-box and target attack scenarios, the *WATERMARK* was performed on a DenseNet + CTC based model trained on a Chinese text image dataset, which includes 3.64 million images and 5,989 unique characters. The adversarial examples crafted by *WATERMARK* are human-eye friendly and with high success probabilities. Besides, some adversarial examples even work on Tesseract in a black-box manner.

The Scene Text Recognition is a standard sequential learning task with a varied-length output. By comparison, Optical Character Recognition is a pipeline process that first segments the word into characters and then recognizes a single character. In Scene Text Recognition tasks, the entire image is directly mapped to a word string. Yuan et al. [144] proposed an adaptive attack to accelerate the adversarial attack through multi-task learning [184], which improves learning efficiency and prediction accuracy by learning multiple objectives from a shared representation. They implemented their attack on the Scene Text Recognition model with the Street View Text [52], ICDAR 2013 [185], and IIIT 5 K-word [186] dataset, and achieved over 99.9% success rate with 3–6 times speedup compared to the Convolutional Recurrent Neural Network (CRNN) [187].

The Visual Question Answering model provides an accurate answer in natural language when given an image and a natural language question about the image. Xu et al. [142] evaluated their method on two models (the MCB model [188] and compositional model N2NMN [189]) on the VQA dataset [190]. They evaluated

[17] https://github.com/tesseract-ocr/tesseract.

Table 12
Summary of Applications in Cross-modal Tasks.

Type	Application	Work	Benchmark Dataset	Victim Model
text-and-vision	image captioning	[141]	MSCOCO	CNN + RNN based model
		[142]	Visual Genome	DenseCap
	optical character recognition	[123]	Hillary Clinton's emails corpus	Tesseract
		[143]	Chinese text image dataset	CTC-based OCR model
	visual question answering	[142]	VQA dataset	MCB, N2NMN
		[145]	VQA dataset	Bottom-Up-Attention and TopDown
	scene text recognition	[144]	Street View Text, ICDAR 2013, IIIT 5 K-word	CRNN
	visual semantic embeddings	[146]	MSCOCO	VSE++
	general V + L task	[147]	COCO, Visual Genome, Conceptual Captions, SBU Captions	UNITER
text-and-audio	speech recognition	[21]	Mozilla Common Voice	DeepSpeech

the success rate (over 90%) and confidence score (above 0.7) of the victim model to predict the target answer. Moreover, they concluded that the attention, bounding box localization, and internal compositional structures are vulnerable to adversarial attacks. Besides, Tang et al. [145] leveraged an encoder-decoder neural machine translation framework and iterative FGSM [191] to generate semantic equivalent adversarial examples of both visual and textual data as the augmented data, which then was utilized for training a visual question answering model using adversarial learning. The model trained with their method obtained 65.16% accuracy on the clean validation dataset, beating its vanilla training counterpart by 1.84%. The adversarial ly-trained model significantly increases accuracy on adversarial examples by 21.55%.

The Visual Semantic Embedding task bridges the natural language and the underlying visual world. In this task, the embedding spaces of both images and descriptive captions are jointly optimized and aligned. Shi et al. [146] performed adversarial attacks on the textual part through three editing operations (replacing nouns in the caption, changing numerals to different ones, detecting the relations and shuffling the non-interchangeable noun phrases or replacing the prepositions). The evaluation on VSE++ model [192] with the MSCOCO dataset showed that, although VSE++ obtains good performance on the original test set, it is vulnerable to caption-specific adversarial attacks.

More generally, Gan et al. [147] proposed *VILLA*, a large-scale adversarial training strategy for vision-and-language representation learning in tasks including Visual Question Answering [193], Visual Commonsense Reasoning [194], Natural Language Visual Reasoning *for Real* (*NLVR*^{2}) [195], Visual Entailment [196], Referring Expression Comprehension [197], and Image-Text Retrieval [198]. *VILLA* first conducts a task-agnostic adversarial pre-training to lift model performance for all downstream tasks uniformly, then implements a task-specific adversarial fine-tuning to enhance the fine-tuned models additionally. Differing from the conventional approaches, *VILLA* adds adversarial perturbations to word embedding and extracted image-region features, respectively. To enable large-scale training, it adopts the "free" adversarial training strategy and combines it with *KL*-divergence-based regularization to promote higher invariance in the embedding space. Relying on standard bottom-up image features only, *VILLA* improves the single-model performance of UNITER-large from 74.02 to 74.87 on visual question answering tasks and from 62.8 to 65.7 on visual commonsense reasoning tasks. With the ensemble, visual question answering performance is boosted to 75.85 additionally.

Text-and-Audio. The Speech Recognition model recognizes and translates spoken language into text automatically. Carlini et al. [21] crafted targeted audio adversarial examples on automatic speech recognition models. Given any natural waveform x, they constructed a perturbation δ that is almost inaudible so that $x + \delta$ was recognized as any desired phrase. They experimented with the Mozilla Common Voice dataset[18] on the DeepSpeech model [199]. Furthermore, they generated targeted adversarial examples with a 100% success rate for each source-target pair with an average perturbation of −31 dB, particularly, the 95% interval for distortion ranged from −15 dB to −45 dB.

4.2.9. Other tasks

Recently, some researchers have proposed adversarial attacks applied to relatively novel NLP tasks. For example, Cheng et al. [148] proposed a framework to generate adversarial agents rather than adversarial examples in an interactive dialogue system under both black-box and white-box settings. Zheng et al. [44] explored the feasibility of generating syntactic adversarial sentences to lead a dependency parser to make mistakes without altering the original syntactic structures. The experiments with a graph-based dependency parser [200] on the English Penn Treebank showed that up to 77% of input examples admit adversarial perturbations.

5. Defense against textual adversarial attack

The wide application of adversarial attacks in the NLP domain makes researchers aware of NLP intelligent systems' potential severe adversarial threats in the real world. To enhance the defensive capability of DNN in NLP tasks and further improve the security of these intelligent systems against adversarial attacks, researchers have proposed numerous strategies, which contain two types: passive and active defense. The passive method detects adversarial input during the inference procedure, while the active method generally improves the robustness of the model when training it.

5.1. Passive defense

As mentioned above, adversarial texts are perceivable and semantic. Thus, checking input is the most straightforward and general passive defense method. According to the type of adversarial attacks that these methods defend, this survey categorizes these defense strategies as two classes, as shown in Table 13.

For character-level attacks, there are some workable misspelling checking tools, such as the Python autocorrect 0.3.0 package [45] and context-aware spelling check service [24]. Additionally, Pruthi et al. [101] designed a word recognition model with three back-off strategies to check misspellings or typos.

For word-level attacks, Mozes et al. [102] observed the frequency differences between words and their substitutes, and then proposed the *Frequency-guided Word Substitutions* (*FGWS*), which is rule-based and model-agnostic. Besides, Zhou et al. [103] focused on determining whether a particular word is a perturbation and

[18] https://commonvoice.mozilla.org/zh-CN/datasets.

Table 13
Summary of Passive Defense Strategies.

Key Idea	Work	Detail
check misspelling	[45]	Python autocorrect 0.3.0 package.
	[24]	A context-aware spelling check service.
	[101]	A model with three back-off strategies.
check word	[102]	Discriminating the perturbated words by frequency differences between words and their substitutions.
	[103]	Detecting adversarial words in a sentence.

proposed *DISP*, which compares embeddings of a word and its potential substitutes.

In general, the above passive defense methods, which focus on checking input, barely consider sentence-level attacks. Furthermore, it is unsuitable for adversarial examples based on other languages like Chinese [201].

5.2. Active defense

Active defense focuses on introducing specific methods in the model training process to enhance robustness. Generally, researchers put forward various methods related to the input data or model itself. Active defense strategies in this survey thus are classified into two categories: adversarial training and representation learning, as shown in Table 14.

5.2.1. Adversarial training

Most defense methods based on adversarial training consider both input and model. Furthermore, adversarial training focuses on enhancing models' tolerance of adversarial examples by adding them to the training dataset and preprocessing them properly. Some researchers directly utilized adversarial texts generated by their attack methods [25,35,24,58,46,38,47]. This adversarial data is crafted according to observed patterns of successful attacks. Hence, these approaches have two problems: 1) extensive adversarial examples need to be prepared in advance, resulting in a massive calculation consumption; 2) the effectiveness of these methods has been proved on blocking certain specific attacks, but not on defending other or unknown attacks.

Therefore, some researchers suggested combining the adversarial example generator and the NLP model, and training them using a GAN-style approach. Kang et al. [104] proposed a knowledge-guided adversarial example generator for incorporating enormous lexical resources in entailment models via only a few rule templates. They proposed the first GAN-style approach for training the entailment model with a sequence-to-sequence model, iteratively improving both the entailment system and the differentiable part of the generator. Likewise, Xu et al. [105] proposed *LexicalAT*, which consists of a generator and a pre-trained sentiment classifier. The generator crafts adversarial texts using *WordNet*. Considering the discreteness of the generator, they trained *LexicalAT* through the policy gradient, which is an RL approach.

Furthermore, to overcome *OOV* words and distribution differences in character-level adversarial examples, Liu et al. [107] propose a framework, which jointly uses character embedding and adversarial stability training, as shown in Fig. 8. For the *OOV* word problem, they expressed the word w_i in sentence s as the character-level word representation $e_i(i = 1, 2, \ldots, n)$ that preserves the information of w_i. For the distribution difference problem, they added some tiny perturbations to each word w_i, and then represented $w_p$in the same way as w_i.

Considering the requirement of diversified adversarial examples, Liu et al. [108] proposed a model-driven approach, which

Table 14
Summary of Active Defense Strategies.

Subcategory	Key Idea	Work	Detail
Adversarial Training	directly use adversarial examples	[25,35,24,58,46,38,47]	Using adversarial texts generated by current attack methods as the training examples straightforwardly.
	special strategy	[104]	Training using a GAN-style approach with a knowledge-guided adversarial example generator for incorporating large lexical resources.
		[105]	Training the generator and the pre-trained sentiment classifier by adopting policy gradient.
		[107]	Jointly using character embedding and adversarial stability training, to overcome *OOV* words and distribution differences.
		[108]	Dynamically crafting various adversarial texts based on parameters of the current model, for the requirement of diversified adversarial examples.
		[109]	Iterative *build it-break it-fix it* strategy with humans and models in the loop, to detect offensive language in the real world.
		[106]	Generating virtual adversarial examples and virtual labels in the embedding space, to reduce calculations.
Representation Learning	randomize input	[110]	Training embedding model on noisy texts.
		[111]	Encoding words with their synonyms selected by a dynamic Random Synonym Substitution algorithm.
		[112]	Randomly sampling embedding vectors for each word in an input sentence from a convex hull spanned by the word and its synonyms to craft virtual sentences.
	unify input representation	[38]	Taking the average character embedding as a word representation.
		[41]	Clustering and encoding all synonyms to a unique code.
		[48]	Encoding words as sequences of symbols and taking the context of a word into account when generating the embedding.
		[49]	Mapping sentences to a smaller and discrete space of encodings.
	design effective representation	[113]	Fine-tuning both local features (word-level representation) and global features (sentence-level representation).
		[114]	Utilizing disentangled representation learning to improve robustness.
		[115]	Linking multi-head attention to structured embeddings.
		[116]	Augmenting input sentences with their corresponding predicate-argument structures.
		[202]	Defining the minimum distance of a text from the decision boundary in the embedding space.

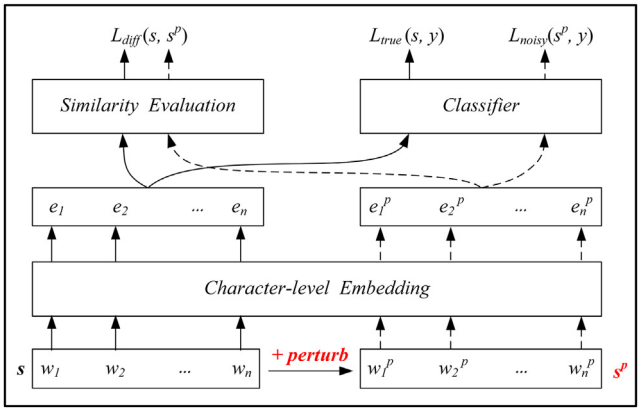

Fig. 8. The framework in [107] to defend character-level adversarial attacks. $L_{\text{diff}}(s, s^p), L_{\text{true}}(S, Y)$ and $L_{\text{noisy}}\left(S^p, Y\right)$ evaluate the similarity between s and its noisy counterpart s^p, the classification accuracy for s and the classification accuracy for s^p, respectively.

dynamically crafts various adversarial texts based on parameters of the current model and further trains the model using these generated texts in an iterative schedule to improve the robustness of machine reading comprehension models. Their training process includes three steps: 1) take a pre-trained machine reading comprehension model as the adversarial generator, and train perturbation embedding sequences to minimize output probabilities of real answers under given questions and passages; 2) greedily sample word sequences from perturbation embeddings as misleading texts to create and enrich the adversarial example set; 3) train the model to maximize probabilities of real answers to block those adversarial examples, then return to step 1 with the retrained model as a new generator.

For detecting the offensive language of dialogue in the real world, Dinan et al. [109] suggested an iterative *build it-break it-fix it* strategy with humans and models in the loop. It includes three steps: 1) Build it: build a model (*A0*) capable of detecting *OFFENSIVE* messages; 2) Break it: ask crowd-workers to try to "beat the system" by submitting messages that *A0* marks as *SAFE* but that the worker considers being *OFFENSIVE*; 3) Fix it: train a new model on these collected examples to make the model more robust to these adversarial attacks, then turn to the step 1.

Considering massive calculations caused by various constraints such as imperceptibility and semantics, Li et al. [106] improved the Virtual Adversarial Training (VAT) [203], which improves efficientness by generating virtual adversarial examples and virtual labels in the embedding space. Without increasing the overhead, the *Token-Aware Virtual Adversarial Training* (*TA-VAT*) crafts fine-grained token-aware perturbations. It introduces a token-level accumulated perturbation vocabulary to initialize the perturbations better and then uses a token-level normalization ball to constrain them pertinently.

Overall, adversarial training strategies have improved models' robustness to a large extent and overcome some problems like adversarial example preparation and calculation consumption. However, with the increase in training iterations and attack types considered, there is a significant possibility that the model robustness is improved against more attacks, while the accuracy decreases resulting from over-fitting adversarial examples.

5.2.2. Representation learning

For the representation learning, researchers mainly properly design the embedding model based on the main ideas of current methods, including three categories: randomizing input, unifying input representation, and designing more effective representation approaches.

For randomizing input, researchers introduce random perturbations to the input during the training step. Unlike adversarial training strategies, the randomized input is with the original label instead of the label with proper modifications. For example, Li et al. [110] proposed to train a special embedding model on noisy texts like tweets, utilizing the randomization characteristics in the dataset. However, typos are unpredictable, and a corpus cannot contain all possible incorrectly spelled versions of a word. Thus, Wang et al. [111] proposed the *Random Substitution Encoding* (*RSE*), which encodes words in the input with their synonyms selected by a dynamic random synonym substitution algorithm. Then, they introduced the *RSE* between the input and the embedding layer, thereby making more labeled neighborhood data for robustness classifiers. Furthermore, Zhou et al. [112] proposed model-agnostic *Dirichlet Neighborhood Ensemble* (*DNE*) to block substitution-based attacks. In the training step, to craft virtual sentences, the *DNE* samples embedding vectors for each word in an input sentence from a convex hull spanned by the word and its synonyms. Then, the generated adversarial sentences are used to train the classifier to improve its robustness. A similar sampling method is introduced in the inference step, and a *CBW-D* ensemble algorithm [204] is adopted to output the final prediction.

For unifying input representation, researchers improved the model's generalization by encoding input and their neighbors with the same representation. Belinkov et al. [38] took the average character embedding as the word representation to deal with adversarial examples generated by character scrambling, such as swapping two letters in a word. Wang et al. [41] designed the *Synonym Encoding Method* (*SEM*), an encoder network that clusters and encodes all synonyms to a unique encoding to force all neighboring words to have an equal representation in the embedding space. Malykh et al. [48] suggested the *Robust Vectors* (*RoVe*), which is an embedding model with two features: open vocabulary and context-dependency. The former encodings words as sequences of symbols to produce embeddings for *OOV* words. The latter takes the context of a word into account when generating the embedding. Furthermore, Jones et al. [49] proposed the *Robust Encodings* (*RobEn*) for mapping sentences to a smaller and discrete space of encodings. Taking adversarial typos as an example, *RobEn* reduced the problem of generating token-level encodings by assigning vocabulary words to clusters and proposed two token-level robust encodings: connected component encodings and agglomerative cluster encodings.

For designing more effective representation approaches, Wang et al. [113] proposed *InfoBERT* to enhance language representations by fine-tuning both local features (word-level representation) and global features (sentence-level representation). *InfoBERT* contains an Information Bottleneck Regularizer extracting approximate minimally sufficient statistics while removing noisy information, and an Anchored Feature Regularizer selecting useful local stable features and maximizing the mutual information between local features and global features. Wu et al. [114] utilized disentangled representation learning to improve the robustness and generality of NLP models. They first mapped an input to a set of representations $\{z_1, z_2, \ldots, z_K\}$. Then, $\{z_1, z_2, \ldots, z_K\}$ were mapped to different logits *ls*, the ensemble of which was used to make the final prediction *y*. In detail, the *L*2regularizer is added on z_s, and Total Correlation is added under the Variational Information Bottleneck framework. Li et al. [115] first linked the multi-head attention [166] to structured embeddings for using each head to linearly project the query and key into a new embedding space. Then, they directly added external knowledge like *WordNet* to this attention mechanism, forcing the model to explore beyond the data distribution of a specific task. Moosavi et al. [116] augmented input sen-

Original: Someone takes the drink, then holds it.

Augmented: Someone takes the drink, then holds it. [PRD] takes [AG0] Someone [AG1] the drink [PRE] [PRD] holds [AG0] Someone [AG1] it [PRE]

Fig. 9. An example of the method [116], which augments an input sentence with its predicate-argument structures. It specified the beginning of the augmentation by the [PRD] token that indicates the subsequent tokens are the detected predicate, and specified the ARG0 and ARG1 arguments with [AG0] and [AG1] tokens, respectively. The [PRE] token specifies that the end of the detected predicate-argument structure is specified by the [PRE] token.

tences with their corresponding predicate-argument structures using the PropBank-style semantic role labeling model [205], to provide a higher-level abstraction over different realizations of the same meaning, as shown in Fig. 9. While La et al. [202] proposed *Maximal Safe Radius* (*MSR*) that defines the minimum distance of a text from the decision boundary in the embedding space. *MSR* was approximated by a lower bound obtained by constraint relaxation techniques and an upper bound gained by the *Monte Carlo Tree Search algorithm*.

Generally, randomizing input and unifying input representation are similar to the passive input checking approach, because both essentially defend against character and word-level attacks. Differently, the former (randomizing input) improves the tolerance of DNN models to adversarial perturbations, similar to the idea of adversarial training. Besides, designing representation pays more attention to the embedding network. It aims to improve the representation ability of the model, so that the model can effectively learn the critical information of an input.

6. Conclusion

This survey comprehensively reviews the research progress of adversarial attack and defense technology in the NLP domain. First, we briefly present the textual adversarial example. Then, we regard the example generation strategy as a novel criterion and propose a two-level classification method for categorizing adversarial attacks on texts by considering both semantic granularity and example generation strategy. Besides, we summarize the applications of adversarial attacks in the NLP field. Finally, we conclude current defense methods against textual adversarial attacks.

It can be found that the advancement of adversarial technology in the CV field promotes its development in the NLP domain. Thus, various textual adversarial attack methods have been proposed in succession. However, the progress of textual adversarial technology is still facing enormous challenges. We conclude five significant challenging issues in the NLP domain from the adversarial attack and defense perspective, as follows:

- **Extra imperceptibility constraints.** Unlike the image data, the textual data is discrete, perceivable, and semantic. These unique characteristics bring significant problems to the implementation of adversarial attacks. Thus, current methods usually make a tradeoff between imperceptibility and attack performance. Therefore, the corresponding research numbers and attack effects are far less than those in the CV field.
- **Practicality in the real world.** When processing the textual input, most attack methods perform greedily. Thus, these methods are significantly more time-consuming, query-consuming, and computationally complex. Such approaches are unsuitable for implementing black-box, real-time, and practical adversarial attacks on intelligence systems in the real world.
- **Limited transferability.** Although researchers have proposed some input-agnostic universal adversarial perturbation generation methods, meaningless triggers could reduce the readability of inputs. Therefore, there are still no well-performed adversarial attack methods, which can fool any DNN. Furthermore, current research focuses on English texts, and only a few studies have considered other languages, such as Chinese. Besides, an approach designed for a certain-language text usually cannot deal with input in other languages.
- **DNN optimization requirements.** Researchers have already proposed several passive and active defense strategies, which consider the model and input data, to handle adversarial attacks. However, most of these methods focus on input checking and representation; few aim to optimize the architecture or objective function of DNN models. Note that improving the robustness of the NLP model itself is an essential way to defend against adversarial attacks.
- **Universal defense.** Existing defense approaches generally aim at a specific type of adversarial attack, similar to those in the CV field. There is still no defense strategy that can defend against all different types of attacks. With the continuous development of different attack methods, it is necessary to investigate and propose a unified model, which can tackle multiple or even all adversarial attacks.

Based on these challenges, combined with the current research status, we point out five potential future directions of research work in textual adversarial attack and defense area, as follows:

- **Pre-trained model-based adversarial attacks.** In recent years, pre-trained models such as BERT [131], GPT-2 [62], and T5 [206] have developed rapidly. Since these pre-trained models extract text and word semantics well, they continuously reach the state-of-the-art in several NLP tasks like machine translation, natural language inference, and text summarization. As mentioned before, designing textual adversarial attack methods requires additional semantic consistency conditions to be considered. Thus, utilizing pre-trained models to handle the semantic consistency requirements in textual adversarial attack methods may be one future direction in this field.
- **Modification operations in the latent feature space.** Unlike image data, improper modification on textual examples usually leads to spelling, grammatical and other errors. Therefore, human eyes and detectors easily detect adversarial texts generated by existing character-level and word-level attack methods. Recently, sentence-level approaches, which extract the text semantics and modify the whole text while keeping the semantics unchanged, have rapidly developed. The adversarial texts crafted by these methods have fewer grammatical and spelling errors but perform poorly in terms of readability and syntactic structure. Therefore, modifying the sentence representation in latent feature space to improve the readability while maintaining semantic consistency will be a future research hotspot.
- **Universal defense approaches.** Researchers have not yet proposed defense methods that can defend against all different types of attacks. In both CV and NLP domains, most of the existing defense strategies are proposed aiming at one or a specific class of attack methods, thus are effective against a small number of types of attacks, but not others. Because of the development trend of defense methods in recent years, defense algorithms based on improving the model itself have gradually become the research focus. Hence, researchers may focus on combining models' characteristics and improving the model architecture, loss functions, and hyper-parameters to defend against a variety of different attacks.

- **Benchmark platform for textual adversarial attack, defense, and evaluation.** In the CV field, researchers have proposed several adversarial attack and defense toolboxs, such as CleverHans [207], Foolbox [208], and Advertorch [209]. However, as far as we know, Textattack [210] and Openattack [211] are the only textual adversarial attack toolbox currently available. Nevertheless, both of them do not cover defense and evaluation functions. Since different attack and defense methods are being proposed and applied to various NLP tasks, it is challenging to compare the advantages and disadvantages of these methods. Therefore, it is essential to develop a textual adversarial benchmark platform that integrates attack, defense, and evaluation functions.
- **Application in emerging tasks.** Currently, textual adversarial attack technology is mainly used to attack various text processing intelligence algorithms to assist researchers in identifying potential security threats in existing intelligence algorithms and improving them. Researchers can vigorously explore novel tasks where adversarial attack technology can be applied in the future. For example, utilizing adversarial technology in the 3D modelling process to improve the realism of 3D models [212,213], applying adversarial examples to human-machine verification [214,215] to fool machines but do not affect regular users.

CRediT authorship contribution statement

Shilin Qiu: Conceptualization, Investigation, Writing – original draft, Writing – review & editing. **Qihe Liu:** Writing – review & editing, Funding acquisition, Project administration. **Shijie Zhou:** Supervision, Project administration. **Wen Huang:** Investigation, Writing – review & editing.

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.

Acknowledgment

This work was supported by the Sichuan Science and Technology Program [Grant Nos. 2019YFG0399, 2020YFG0472, 2020YFG0031].

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90.
[2] L. Qin, N. Yu, D. Zhao, Applying the convolutional neural network deep learning technology to behavioural recognition in intelligent video, Tehnički vjesnik 25 (2) (2018) 528–535.
[3] M.S. Hossain, G. Muhammad, Emotion recognition using deep learning approach from audio–visual emotional big data, Inf. Fusion 49 (2019) 69–78.
[4] A. Chatterjee, U. Gupta, M.K. Chinnakotla, R. Srikanth, M. Galley, P. Agrawal, Understanding emotions in text using deep learning and big data, Comput. Hum. Behav. 93 (2019) 309–317.
[5] W. Guo, H. Gao, J. Shi, B. Long, L. Zhang, B.-C. Chen, D. Agarwal, Deep natural language processing for search and recommender systems, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 3199–3200.
[6] L. Yang, Y. Li, J. Wang, R.S. Sherratt, Sentiment analysis for e-commerce product reviews in chinese based on sentiment lexicon and deep learning, IEEE Access 8 (2020) 23522–23530.
[7] B. Sisman, J. Yamagishi, S. King, H. Li, An overview of voice conversion and its challenges: From statistical modeling to deep learning, IEEE/ACM Transactions on Audio, Speech, and Language Processing.
[8] M. Saravanan, B. Selvababu, A. Jayan, A. Anand, A. Raj, Arduino based voice controlled robot vehicle, in: IOP Conference Series: Materials Science and Engineering, Vol. 993, IOP Publishing, 2020, p. 012125.
[9] H. Liu, T. Fang, T. Zhou, Y. Wang, L. Wang, Deep learning-based multimodal control interface for human-robot collaboration, Procedia CIRP 72 (2018) 3–8.
[10] C.-S. Oh, J.-M. Yoon, Hardware acceleration technology for deep-learning in autonomous vehicles, in: 2019 IEEE International Conference on Big Data and Smart Computing (BigComp) IEEE, 2019, pp. 1–3.
[11] M. Coccia, Deep learning technology for improving cancer care in society: New directions in cancer imaging driven by artificial intelligence, Technol. Soc. 60 (2020) 101198.
[12] J. Harikrishnan, A. Sudarsan, A. Sadashiv, R.A. Ajai, Vision-face recognition attendance monitoring system for surveillance using deep learning technology and computer vision, in: 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), IEEE, 2019, pp. 1–5.
[13] S. So, J. Mun, J. Rho, Simultaneous inverse design of materials and structures via deep learning: demonstration of dipole resonance engineering using core–shell nanoparticles, ACS Appl. Mater. Interfaces 11 (27) (2019) 24264–24268.
[14] H.-P. Chan, L.M. Hadjiiski, R.K. Samala, Computer-aided diagnosis in the era of deep learning, Med. Phys. 47 (5) (2020) e218–e227.
[15] F. Zhang, P.P. Chan, B. Biggio, D.S. Yeung, F. Roli, Adversarial feature selection against evasion attacks, IEEE Trans. Cybern. 46 (3) (2015) 766–777.
[16] K.D. Julian, J. Lopez, J.S. Brush, M.P. Owen, M.J. Kochenderfer, Policy compression for aircraft collision avoidance systems IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), IEEE 2016 (2016) 1–10.
[17] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, arXiv preprint arXiv:1312.6199.
[18] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, D. Song, Robust physical-world attacks on deep learning visual classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1625–1634.
[19] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572.
[20] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, A. Yuille, Adversarial examples for semantic segmentation and object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1369–1378.
[21] N. Carlini, D. Wagner, Audio adversarial examples: Targeted attacks on speech-to-text, in IEEE Security and Privacy Workshops (SPW), IEEE 2018 (2018) 1–7.
[22] H. Yakura, J. Sakuma, Robust audio adversarial example for a physical attack, arXiv preprint arXiv:1810.11793.
[23] R. Taori, A. Kamsetty, B. Chu, N. Vemuri, Targeted adversarial examples for black box audio systems, 2019 IEEE Security and Privacy Workshops (SPW), IEEE 2019 (2019) 15–20.
[24] J. Li, S. Ji, T. Du, B. Li, T. Wang, Textbugger: Generating adversarial text against real-world applications, arXiv preprint arXiv:1812.05271.
[25] J. Ebrahimi, A. Rao, D. Lowd, D. Dou, Hotflip: White-box adversarial examples for text classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 31–36, 10.18653/v1/P18-2006.
[26] X. Liu, Y. Lin, H. Li, J. Zhang, Adversarial examples: Attacks on machine learning-based malware visualization detection methods, arXiv preprint arXiv:1808.01546 10 (3326285.3329073).
[27] J. Chen, Z. Yang, D. Yang, Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification, arXiv preprint arXiv:2004.12239.
[28] D. Mekala, J. Shang, Contextualized weak supervision for text classification, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 323–333.
[29] R.K. Bakshi, N. Kaur, R. Kaur, G. Kaur, Opinion mining and sentiment analysis, in: 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), 2016, pp. 452–455.
[30] P. Gupta, V. Gupta, A survey of text question answering techniques, International Journal of Computer Applications 53 (4).
[31] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., Google's neural machine translation system: Bridging the gap between human and machine translation, arXiv preprint arXiv:1609.08144.
[32] Y. Duan, C. Xu, J. Pei, J. Han, C. Li, Pre-train and plug-in: Flexible conditional text generation with variational auto-encoders, arXiv preprint arXiv:1911.03882.
[33] Y. Tay, D. Bahri, C. Zheng, C. Brunk, D. Metzler, A. Tomkins, Reverse engineering configurations of neural text generation models, arXiv preprint arXiv:2004.06201.
[34] N. Papernot, P. McDaniel, A. Swami, R. Harang, Crafting adversarial input sequences for recurrent neural networks, MILCOM 2016–2016 IEEE Military Communications Conference, IEEE (2016) 49–54.
[35] J. Ebrahimi, D. Lowd, D. Dou, On adversarial examples for character-level neural machine translation, in: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2018, pp. 653–663.
[36] C. Wong, Dancin seq2seq: Fooling text classifiers with adversarial text example generation, arXiv preprint arXiv:1712.05419.

[37] Y. Zang, B. Hou, F. Qi, Z. Liu, X. Meng, M. Sun, Learning to attack: Towards textual adversarial attacking in real-world situations, arXiv preprint arXiv:2009.09192.
[38] Y. Belinkov, Y. Bisk, Synthetic and natural noise both break neural machine translation, arXiv preprint arXiv:1711.02173.
[39] S. Eger, Y. Benz, From hero to z\)éroe: A benchmark of low-level adversarial attacks, arXiv preprint arXiv:2010.05648.
[40] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, K.-W. Chang, Generating natural language adversarial examples, arXiv preprint arXiv:1804.07998.
[41] X. Wang, H. Jin, K. He, Natural language adversarial attacks and defenses in word level, arXiv preprint arXiv:1909.06723.
[42] Z. Shao, Z. Liu, J. Zhang, Z. Wu, M. Huang, Advexpander: Generating natural language adversarial examples by expanding text, arXiv preprint arXiv:2012.10235.
[43] L. Xu, I. Ramirez, K. Veeramachaneni, Rewriting meaningful sentences via conditional bert sampling and an application on fooling text classifiers, arXiv preprint arXiv:2010.11869.
[44] X. Zheng, J. Zeng, Y. Zhou, C.-J. Hsieh, M. Cheng, X.-J. Huang, Evaluating and enhancing the robustness of neural network-based dependency parsing models with adversarial examples, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 6600–6610.
[45] J. Gao, J. Lanchantin, M.L. Soffa, Y. Qi, Black-box generation of adversarial text sequences to evade deep learning classifiers, in IEEE Security and Privacy Workshops (SPW), IEEE 2018 (2018) 50–56.
[46] Y. Wang, M. Bansal, Robust machine comprehension models via adversarial training, arXiv preprint arXiv:1804.06473.
[47] Y. Zang, F. Qi, C. Yang, Z. Liu, M. Zhang, Q. Liu, M. Sun, Word-level textual adversarial attacking as combinatorial optimization, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 6066–6080.
[48] V. Malykh, Robust to noise models in natural language processing tasks, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, 2019, pp. 10–16.
[49] E. Jones, R. Jia, A. Raghunathan, P. Liang, Robust encodings: A framework for combating adversarial typos, arXiv preprint arXiv:2005.01229.
[50] J. Gilmer, R.P. Adams, I. Goodfellow, D. Andersen, G.E. Dahl, Motivating the rules of the game for adversarial example research, arXiv preprint arXiv:1807.06732.
[51] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, D. Mukhopadhyay, Adversarial attacks and defences: A survey, arXiv preprint arXiv:1810.00069.
[52] X. Yuan, P. He, Q. Zhu, X. Li, Adversarial examples: Attacks and defenses for deep learning, IEEE transactions on neural networks and learning systems 30 (9) (2019) 2805–2824.
[53] J. Zhang, C. Li, Adversarial examples: Opportunities and challenges, IEEE transactions on neural networks and learning systems 31 (7) (2019) 2578–2593.
[54] S. Qiu, Q. Liu, S. Zhou, C. Wu, Review of artificial intelligence adversarial attack and defense technologies, Applied Sciences 9 (5) (2019) 909.
[55] W. Wang, L. Wang, R. Wang, Z. Wang, A. Ye, Towards a robust deep neural network in texts: A survey, arXiv preprint arXiv:1902.07285.
[56] W.E. Zhang, Q.Z. Sheng, A. Alhazmi, C. Li, Adversarial attacks on deep-learning models in natural language processing: A survey, ACM Transactions on Intelligent Systems and Technology (TIST) 11 (3) (2020) 1–41.
[57] A. Huq, M.T. Pervin, Adversarial attacks and defense on texts: A survey, arXiv e-prints, 2020, arXiv–2005..
[58] R. Jia, P. Liang, Adversarial examples for evaluating reading comprehension systems, arXiv preprint arXiv:1707.07328.
[59] N.J. Nizar, A. Kobren, Leveraging extracted model adversaries for improved black box attacks, arXiv preprint arXiv:2010.16336.
[60] Y. Gil, Y. Chai, O. Gorodissky, J. Berant, White-to-black: Efficient distillation of black-box adversarial attacks, arXiv preprint arXiv:1904.02405.
[61] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: International conference on machine learning, PMLR, 2014, pp. 1188–1196.
[62] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, OpenAI blog 1 (8) (2019) 9.
[63] M. Iyyer, J. Wieting, K. Gimpel, L. Zettlemoyer, Adversarial example generation with syntactically controlled paraphrase networks, arXiv preprint arXiv:1804.06059.
[64] M. Cheng, J. Yi, P.-Y. Chen, H. Zhang, C.-J. Hsieh, Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples., AAAI (2020) 3601–3608.
[65] B. Liang, H. Li, M. Su, P. Bian, X. Li, W. Shi, Deep text classification can be fooled, arXiv preprint arXiv:1704.08006.
[66] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, Advances in neural information processing systems 26 (2013) 3111–3119.
[67] K. Taga, K. Kameyama, K. Toraichi, Regularization of hidden layer unit response for neural networks, in: 2003 IEEE Pacific Rim Conference on Communications Computers and Signal Processing (PACRIM 2003)(Cat. No. 03CH37490), Vol. 1, IEEE, 2003, pp. 348–351.
[68] T. Tanay, L. Griffin, A boundary tilting persepective on the phenomenon of adversarial examples, arXiv preprint arXiv:1608.07690.
[69] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, A. Madry, Adversarial examples are not bugs, they are features, Advances in Neural Information Processing Systems (2019) 125–136.
[70] P. Michel, X. Li, G. Neubig, J.M. Pino, On evaluation of adversarial perturbations for sequence-to-sequence models, arXiv preprint arXiv:1903.06620.
[71] R. Maheshwary, S. Maheshwary, V. Pudi, Generating natural language attacks in a hard label black box setting, arXiv preprint arXiv:2012.14956.
[72] A. Mathai, S. Khare, S. Tamilselvam, S. Mani, Adversarial black-box attacks on text classifiers using multi-objective genetic optimization guided by deep networks, arXiv preprint arXiv:2011.03901.
[73] L. Yuan, X. Zheng, Y. Zhou, C.-J. Hsieh, K.-W. Chang, X. Huang, Generating universal language adversarial examples by understanding and enhancing the transferability across neural models, arXiv preprint arXiv:2011.08558.
[74] E.J. Anderson, M.C. Ferris, Genetic algorithms for combinatorial optimization: the assemble line balancing problem, ORSA Journal on Computing 6 (2) (1994) 161–173.
[75] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4, IEEE, 1995, pp. 1942–1948.
[76] S. Tan, S. Joty, M.-Y. Kan, R. Socher, It's morphin'time! combating linguistic discrimination with inflectional perturbations, arXiv preprint arXiv:2005.04364.
[77] N. Xu, O. Feyisetan, A. Aggarwal, Z. Xu, N. Teissier, Differentially private adversarial robustness through randomized perturbations, arXiv preprint arXiv:2009.12718.
[78] S. Samanta, S. Mehta, Towards crafting text adversarial samples, arXiv preprint arXiv:1707.02812.
[79] D. Jin, Z. Jin, J.T. Zhou, P. Szolovits, 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, Vol. 34, 2020, pp. 8018–8025.
[80] R. Maheshwary, S. Maheshwary, V. Pudi, A context aware approach for generating natural language attacks, arXiv preprint arXiv:2012.13339.
[81] S. Ren, Y. Deng, K. He, W. Che, Generating natural language adversarial examples through probability weighted word saliency, in: Proceedings of the 57th annual meeting of the association for computational linguistics, 2019, pp. 1085–1097.
[82] M. Hossam, T. Le, H. Zhao, D. Phung, Explain2attack: Text adversarial attacks via cross-domain interpretability.
[83] P. Yang, J. Chen, C.-J. Hsieh, J.-L. Wang, M.I. Jordan, Greedy attack and gumbel attack: Generating adversarial examples for discrete data, Journal of Machine Learning Research 21 (43) (2020) 1–36.
[84] H. Zhang, H. Zhou, N. Miao, L. Li, Generating fluent adversarial examples for natural languages, arXiv preprint arXiv:2007.06174.
[85] D. Li, Y. Zhang, H. Peng, L. Chen, C. Brockett, M.-T. Sun, B. Dolan, Contextualized perturbation for textual adversarial attack, arXiv preprint arXiv:2009.07502.
[86] D. Emelin, I. Titov, R. Sennrich, Detecting word sense disambiguation biases in machine translation for model-agnostic adversarial attacks, arXiv preprint arXiv:2011.01846.
[87] M. Behjati, S.-M. Moosavi-Dezfooli, M.S. Baghshah, P. Frossard, Universal adversarial attacks on text classifiers, in: ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) IEEE, 2019, pp. 7345–7349.
[88] L. Song, X. Yu, H.-T. Peng, K. Narasimhan, Universal adversarial attacks with natural triggers for text classification, arXiv preprint arXiv:2005.00174.
[89] E. Wallace, S. Feng, N. Kandpal, M. Gardner, S. Singh, Universal adversarial triggers for attacking and analyzing nlp, arXiv preprint arXiv:1908.07125.
[90] P. Atanasova, D. Wright, I. Augenstein, Generating label cohesive and well-formed adversarial claims, arXiv preprint arXiv:2009.08205.
[91] M.T. Ribeiro, S. Singh, C. Guestrin, Semantically equivalent adversarial rules for debugging nlp models, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018, pp. 856–865.
[92] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473.
[93] A. See, P.J. Liu, C.D. Manning, Get to the point: Summarization with pointer-generator networks, arXiv preprint arXiv:1704.04368.
[94] Z. Zhao, D. Dua, S. Singh, Generating natural adversarial examples, arXiv preprint arXiv:1710.11342.
[95] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Advances in neural information processing systems 27.
[96] R.S. Sutton, A.G. Barto, Reinforcement learning, An introduction, 2011.
[97] T. Wang, X. Wang, Y. Qin, B. Packer, K. Li, J. Chen, A. Beutel, E. Chi, Cat-gen: Improving robustness in nlp models via controlled adversarial text generation, arXiv preprint arXiv:2010.02338.
[98] T. Niu, M. Bansal, Adversarial over-sensitivity and over-stability strategies for dialogue models, arXiv preprint arXiv:1809.02079.
[99] M. Blohm, G. Jagfeld, E. Sood, X. Yu, N.T. Vu, Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension, in: Proceedings of the 22nd Conference on Computational Natural Language Learning, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 108–118, 10.18653/v1/K18-1011.
[100] P. Vijayaraghavan, D. Roy, Generating black-box adversarial examples for text classifiers using a deep reinforced model, in: Joint European Conference on

Machine Learning and Knowledge Discovery in Databases, Springer, 2019, pp. 711–726.
[101] D. Pruthi, B. Dhingra, Z.C. Lipton, Combating adversarial misspellings with robust word recognition, arXiv preprint arXiv:1905.11268.
[102] M. Mozes, P. Stenetorp, B. Kleinberg, L.D. Griffin, Frequency-guided word substitutions for detecting textual adversarial examples, arXiv preprint arXiv:2004.05887.
[103] Y. Zhou, J.-Y. Jiang, K.-W. Chang, W. Wang, Learning to discriminate perturbations for blocking adversarial attacks in text classification, arXiv preprint arXiv:1909.03084.
[104] D. Kang, T. Khot, A. Sabharwal, E. Hovy, Adventure: Adversarial training for textual entailment with knowledge-guided examples, arXiv preprint arXiv:1805.04680.
[105] J. Xu, L. Zhao, H. Yan, Q. Zeng, Y. Liang, S. Xu, Lexicalat: Lexical-based adversarial reinforcement training for robust sentiment classification, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 5521–5530.
[106] L. Li, X. Qiu, Textat: Adversarial training for natural language understanding with token-level perturbation, arXiv preprint arXiv:2004.14543.
[107] H. Liu, Y. Zhang, Y. Wang, Z. Lin, Y. Chen, Joint character-level word embedding and adversarial stability training to defend adversarial text, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 8384–8391.
[108] K. Liu, X. Liu, A. Yang, J. Liu, J. Su, S. Li, Q. She, A robust adversarial training approach to machine reading comprehension, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 8392–8400.
[109] E. Dinan, S. Humeau, B. Chintagunta, J. Weston, Build it break it fix it for dialogue safety: Robustness from adversarial human attack, arXiv preprint arXiv:1908.06083.
[110] Q. Li, S. Shah, X. Liu, A. Nourbakhsh, Data sets: Word embeddings learned from tweets and general data, in: Proceedings of the International AAAI Conference on Web and Social Media, Vol. 11, 2017.
[111] Z. Wang, H. Wang, Defense of word-level adversarial attacks via random substitution encoding, in: International Conference on Knowledge Science, Engineering and Management, Springer, 2020, pp. 312–324.
[112] Y. Zhou, X. Zheng, C.-J. Hsieh, K.-W. Chang, X. Huang, Defense against adversarial attacks in nlp via dirichlet neighborhood ensemble, arXiv preprint arXiv:2006.11627.
[113] B. Wang, S. Wang, Y. Cheng, Z. Gan, R. Jia, B. Li, J. Liu, Infobert: Improving robustness of language models from an information theoretic perspective, arXiv preprint arXiv:2010.02329.
[114] J. Wu, X. Li, X. Ao, Y. Meng, F. Wu, J. Li, Improving robustness and generality of nlp models using disentangled representations, arXiv preprint arXiv:2009.09587.
[115] A.H. Li, A. Sethy, Knowledge enhanced attention for robust natural language inference, arXiv preprint arXiv:1909.00102.
[116] N.S. Moosavi, M. de Boer, P.A. Utama, I. Gurevych, Improving robustness by augmenting training sentences with predicate-argument structures, arXiv preprint arXiv:2010.12510.
[117] M. Kusner, Y. Sun, N. Kolkin, K. Weinberger, From word embeddings to document distances, in: International conference on machine learning, 2015, pp. 957–966.
[118] P. Minervini, S. Riedel, Adversarially regularising neural nli models to integrate logical background knowledge, arXiv preprint arXiv:1808.08609.
[119] Y. Cheng, L. Jiang, W. Macherey, Robust neural machine translation with doubly adversarial inputs, arXiv preprint arXiv:1906.02443.
[120] V. Kuleshov, S. Thakoor, T. Lau, S. Ermon, Adversarial examples for natural language classification problems.
[121] M. Sato, J. Suzuki, H. Shindo, Y. Matsumoto, Interpretable adversarial perturbation in input embedding space for text, arXiv preprint arXiv:1805.02917.
[122] Z. Gong, W. Wang, B. Li, D. Song, W.-S. Ku, Adversarial texts with gradient methods, arXiv preprint arXiv:1801.07175.
[123] C. Song, V. Shmatikov, Fooling ocr systems with adversarial text images, arXiv preprint arXiv:1802.05385.
[124] S.-M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, Deepfool: a simple and accurate method to fool deep neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2574–2582.
[125] G.A. Miller, Wordnet: a lexical database for english, Commun. ACM 38 (11) (1995) 39–41.
[126] J. Zhao, Y. Kim, K. Zhang, A. Rush, Y. LeCun, Adversarially regularized autoencoders, in: International conference on machine learning PMLR, 2018, pp. 5902–5911.
[127] K. Krishna, G.S. Tomar, A.P. Parikh, N. Papernot, M. Iyyer, Thieves on sesame street! model extraction of bert-based apis.
[128] K. Sohn, H. Lee, X. Yan, Learning structured output representation using deep conditional generative models, Advances in neural information processing systems 28 (2015) 3483–3491.
[129] H. Chen, S. Huang, D. Chiang, X. Dai, J. Chen, Combining character and word information in neural machine translation using a multi-level attention, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 1284–1293.
[130] S.J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, V. Goel, Self-critical sequence training for image captioning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7008–7024.
[131] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805.
[132] L. Li, R. Ma, Q. Guo, X. Xue, X. Qiu, Bert-attack: Adversarial attack against bert using bert, arXiv preprint arXiv:2004.09984.
[133] P. Neekhara, S. Hussain, S. Dubnov, F. Koushanfar, Adversarial reprogramming of sequence classification neural networks, CoRR abs/1809.01829.
[134] A. de Wynter, Mischief: A simple black-box attack against transformer architectures, arXiv preprint arXiv:2010.08542.
[135] T. Le, S. Wang, D. Lee, Malcom: Generating malicious comments to attack neural fake news detection models, arXiv preprint arXiv:2009.01048.
[136] Y. Wu, D. Bamman, S. Russell, Adversarial training for relation extraction, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 1778–1783.
[137] G. Bekoulis, J. Deleu, T. Demeester, C. Develder, Adversarial training for multi-context joint entity and relation extraction, arXiv preprint arXiv:1808.06876.
[138] M. Cettolo, N. Jan, S. Sebastian, L. Bentivogli, R. Cattoni, M. Federico, The iwslt 2016 evaluation campaign, in: International Workshop on Spoken Language Translation, 2016.
[139] M. Yasunaga, J. Kasai, D. Radev, Robust multilingual part-of-speech tagging via adversarial training, arXiv preprint arXiv:1711.04903.
[140] W. Han, L. Zhang, Y. Jiang, K. Tu, Adversarial attack and defense of structured prediction models, arXiv preprint arXiv:2010.01610.
[141] H. Chen, H. Zhang, P.-Y. Chen, J. Yi, C.-J. Hsieh, Attacking visual language grounding with adversarial examples: A case study on neural image captioning, arXiv preprint arXiv:1712.02051.
[142] X. Xu, X. Chen, C. Liu, A. Rohrbach, T. Darrell, D. Song, Fooling vision and language models despite localization and attention mechanism, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4951–4961.
[143] L. Chen, W. Xu, Attacking optical character recognition (ocr) systems with adversarial watermarks, arXiv preprint arXiv:2002.03095.
[144] X. Yuan, P. He, X. Lit, D. Wu, Adaptive adversarial attack on scene text recognition, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) IEEE, 2020, pp. 358–363.
[145] R. Tang, C. Ma, W.E. Zhang, Q. Wu, X. Yang, Semantic equivalent adversarial data augmentation for visual question answering, European Conference on Computer Vision, Springer (2020) 437–453.
[146] H. Shi, J. Mao, T. Xiao, Y. Jiang, J. Sun, Learning visually-grounded semantics from contrastive adversarial samples, arXiv preprint arXiv:1806.10348.
[147] Z. Gan, Y.-C. Chen, L. Li, C. Zhu, Y. Cheng, J. Liu, Large-scale adversarial training for vision-and-language representation learning, arXiv preprint arXiv:2006.06195.
[148] M. Cheng, W. Wei, C.-J. Hsieh, Evaluating and enhancing the robustness of dialogue systems: A case study on a negotiation agent, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 3325–3335.
[149] Y. Kim, Y. Jernite, D. Sontag, A. Rush, Character-aware neural language models, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 30, 2016.
[150] M. Schuster, K.K. Paliwal, Bidirectional recurrent neural networks, IEEE transactions on Signal Processing 45 (11) (1997) 2673–2681.
[151] K.S. Tai, R. Socher, C.D. Manning, Improved semantic representations from tree-structured long short-term memory networks, arXiv preprint arXiv:1503.00075.
[152] X. Zhang, J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, Advances in neural information processing systems 28 (2015) 649–657.
[153] Y. Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882.
[154] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.
[155] T. Miyato, A.M. Dai, I. Goodfellow, Adversarial training methods for semi-supervised text classification, arXiv preprint arXiv:1605.07725.
[156] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, A. Bordes, Supervised learning of universal sentence representations from natural language inference data, arXiv preprint arXiv:1705.02364.
[157] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692.
[158] K. Shu, L. Cui, S. Wang, D. Lee, H. Liu, defend: Explainable fake news detection, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 395–405.
[159] D. Zeng, K. Liu, S. Lai, G. Zhou, J. Zhao, Relation classification via convolutional deep neural network, in: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, 2014, pp. 2335–2344.
[160] K. Cho, B. Van Merriënboer, D. Bahdanau, Y. Bengio, On the properties of neural machine translation: Encoder-decoder approaches, arXiv preprint arXiv:1409.1259.

[161] G. Bekoulis, J. Deleu, T. Demeester, C. Develder, Joint entity recognition and relation extraction as a multi-head selection problem, Expert Syst. Appl. 114 (2018) 34–45.
[162] M. Kaneko, Y. Sakaizawa, M. Komachi, Grammatical error detection using error-and grammaticality-specific word embeddings, in: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2017, pp. 40–48.
[163] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, arXiv preprint arXiv:1802.05365.
[164] J. Lee, K. Cho, T. Hofmann, Fully character-level neural machine translation without explicit segmentation, Transactions of the Association for, Computational Linguistics 5 (2017) 365–378.
[165] R. Sennrich, O. Firat, K. Cho, A. Birch, B. Haddow, J. Hitschler, M. Junczys-Dowmunt, S. Läubli, A.V.M. Barone, J. Mokry, et al., Nematus: a toolkit for neural machine translation, arXiv preprint arXiv:1703.04357.
[166] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.
[167] M.-T. Luong, H. Pham, C.D. Manning, Effective approaches to attention-based neural machine translation, arXiv preprint arXiv:1508.04025.
[168] J. Gehring, M. Auli, D. Grangier, Y.N. Dauphin, A convolutional encoder model for neural machine translation, arXiv preprint arXiv:1611.02344.
[169] M. Seo, A. Kembhavi, A. Farhadi, H. Hajishirzi, Bidirectional attention flow for machine comprehension, arXiv preprint arXiv:1611.01603.
[170] S. Wang, J. Jiang, Machine comprehension using match-lstm and answer pointer, arXiv preprint arXiv:1608.07905.
[171] A.W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, Q.V. Le, Qanet: Combining local convolution with global self-attention for reading comprehension, arXiv preprint arXiv:1804.09541.
[172] G. Lample, A. Conneau, Cross-lingual language model pretraining, arXiv preprint arXiv:1901.07291.
[173] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R.R. Salakhutdinov, Q.V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, in: Advances in neural information processing systems, 2019, pp. 5753–5763.
[174] Q. Chen, X. Zhu, Z. Ling, S. Wei, H. Jiang, D. Inkpen, Enhanced lstm for natural language inference, arXiv preprint arXiv:1609.06038.
[175] M. Marcus, B. Santorini, M.A. Marcinkiewicz, Building a large annotated corpus of english: The penn treebank.
[176] J. Nivre, Ž. Agić, M.J. Aranzabe, M. Asahara, A. Atutxa, M. Ballesteros, J. Bauer, K. Bengoetxea, R.A. Bhat, C. Bosco, et al., Universal dependencies 1.2.
[177] R. Lowe, N. Pow, I. Serban, J. Pineau, The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems, arXiv preprint arXiv:1506.08909.
[178] H. He, A. Balakrishnan, M. Eric, P. Liang, Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings, arXiv preprint arXiv:1704.07130.
[179] I. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. Courville, Y. Bengio, A hierarchical latent variable encoder-decoder model for generating dialogues, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 31, 2017.
[180] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, D. Jurafsky, Deep reinforcement learning for dialogue generation, arXiv preprint arXiv:1606.01541.
[181] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: Common objects in context, in: European conference on computer vision, Springer, 2014, pp. 740–755.
[182] J. Johnson, A. Karpathy, L. Fei-Fei, Densecap: Fully convolutional localization networks for dense captioning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 4565–4574.
[183] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D.A. Shamma, et al., Visual genome: Connecting language and vision using crowdsourced dense image annotations, International journal of computer vision 123 (1) (2017) 32–73.
[184] A. Kendall, Y. Gal, R. Cipolla, Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7482–7491.
[185] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L.G. i Bigorda, S.R. Mestre, J. Mas, D.F. Mota, J.A. Almazan, L.P. De Las Heras, Icdar 2013 robust reading competition, in: 2013 12th International Conference on Document Analysis and Recognition, IEEE, 2013, pp. 1484–1493.
[186] A. Mishra, K. Alahari, C. Jawahar, Scene text recognition using higher order language priors, 2012.
[187] B. Shi, X. Bai, C. Yao, An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition, IEEE transactions on pattern analysis and machine intelligence 39 (11) (2016) 2298–2304.
[188] A. Fukui, D.H. Park, D. Yang, A. Rohrbach, T. Darrell, M. Rohrbach, Multimodal compact bilinear pooling for visual question answering and visual grounding, arXiv preprint arXiv:1606.01847.
[189] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, K. Saenko, Learning to reason: End-to-end module networks for visual question answering, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 804–813.
[190] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, D. Parikh, Vqa: Visual question answering, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 2425–2433.
[191] A. Kurakin, I. Goodfellow, S. Bengio, et al., Adversarial examples in the physical world (2016).
[192] F. Faghri, D.J. Fleet, J.R. Kiros, S. Fidler, Vse++: Improving visual-semantic embeddings with hard negatives, arXiv preprint arXiv:1707.05612.
[193] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, D. Parikh, Making the v in vqa matter: Elevating the role of image understanding in visual question answering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6904–6913.
[194] R. Zellers, Y. Bisk, A. Farhadi, Y. Choi, From recognition to cognition: Visual commonsense reasoning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6720–6731.
[195] A. Suhr, S. Zhou, A. Zhang, I. Zhang, H. Bai, Y. Artzi, A corpus for reasoning about natural language grounded in photographs, arXiv preprint arXiv:1811.00491.
[196] N. Xie, F. Lai, D. Doran, A. Kadav, Visual entailment: A novel task for fine-grained image understanding, arXiv preprint arXiv:1901.06706.
[197] L. Yu, P. Poirson, S. Yang, A.C. Berg, T.L. Berg, Modeling context in referring expressions, European Conference on Computer Vision, Springer (2016) 69–85.
[198] K.-H. Lee, X. Chen, G. Hua, H. Hu, X. He, Stacked cross attention for image-text matching, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 201–216.
[199] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, et al., Deep speech: Scaling up end-to-end speech recognition, arXiv preprint arXiv:1412.5567.
[200] T. Dozat, C.D. Manning, Deep biaffine attention for neural dependency parsing, arXiv preprint arXiv:1611.01734.
[201] W. Wang, R. Wang, L. Wang, B. Tang, Adversarial examples generation approach for tendency classification on chinese texts, Ruan Jian Xue Bao/J. Softw. 30 (8) (2019) 2415–2427.
[202] E. La Malfa, M. Wu, L. Laurenti, B. Wang, A. Hartshorn, M. Kwiatkowska, Assessing robustness of text classification through maximal safe radius computation, arXiv preprint arXiv:2010.02004.
[203] T. Miyato, S.-I. Maeda, M. Koyama, S. Ishii, Virtual adversarial training: a regularization method for supervised and semi-supervised learning, IEEE transactions on pattern analysis and machine intelligence 41 (8) (2018) 1979–1993.
[204] A. Dubey, L. v. d. Maaten, Z. Yalniz, Y. Li, D. Mahajan, Defense against adversarial images using web-scale nearest-neighbor search, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8767–8776.
[205] P. Shi, J. Lin, Simple bert models for relation extraction and semantic role labeling, arXiv preprint arXiv:1904.05255.
[206] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, arXiv preprint arXiv:1910.10683.
[207] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, et al., Technical report on the cleverhans v2. 1.0 adversarial examples library, arXiv preprint arXiv:1610.00768.
[208] J. Rauber, W. Brendel, M. Bethge, Foolbox: A python toolbox to benchmark the robustness of machine learning models, arXiv preprint arXiv:1707.04131.
[209] G.W. Ding, L. Wang, X. Jin, Advertorch v0. 1: An adversarial robustness toolbox based on pytorch, arXiv preprint arXiv:1902.07623.
[210] J.X. Morris, E. Lifland, J.Y. Yoo, Y. Qi, Textattack: A framework for adversarial attacks in natural language processing.
[211] G. Zeng, F. Qi, Q. Zhou, T. Zhang, Z. Ma, B. Hou, Y. Zang, Z. Liu, M. Sun, Openattack: An open-source textual adversarial attack toolkit, arXiv preprint arXiv:2009.09191.
[212] Y. Liang, F. He, X. Zeng, J. Luo, An improved loop subdivision to coordinate the smoothness and the number of faces via multi-objective optimization, Integrated Computer-Aided Engineering (Preprint) (2021) 1–19.
[213] A. Lahav, A. Tal, Meshwalker: Deep mesh understanding by random walks, ACM Transactions on Graphics (TOG) 39 (6) (2020) 1–13.
[214] M.I. Hossen, X. Hei, aaecaptcha: The design and implementation of audio adversarial captcha, arXiv preprint arXiv:2203.02735.
[215] M. Kumar, M. Jindal, M. Kumar, Design of innovative captcha for hindi language, Neural Comput. Appl. (2022) 1–36.

Shilin Qiu received the B.E. degree in software engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2017, where she is currently pursuing the Doctoral degree in software engineering.
Her current research interests include deep learning, artificial intelligence adversarial technology.

Qihe Liu received the Ph.D. degree in computer application technology from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2005. He is currently a Associate Professor with the School of Information and Software Engineering, UESTC.
His current research interests include communication and security in network security, machine learning, and artificial intelligence adversarial technology.

Shijie Zhou received the Ph.D. degree in computer science and technology from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2004. He is currently a Professor with the School of Information and Software Engineering, UESTC. His current research interests include communication and security in network security, artificial intelligence, and intelligent transportation.

Wen Huang received the B.E. degree in computer science and technology from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2016, where he is currently pursuing the Doctoral degree in software engineering. His current research interests include cryptography and differential privacy.