PRAGMATIC EVALUATION OF ADVERSARIAL EXAMPLES IN NATURAL LANGUAGE

Anonymous authors
Paper under double-blind review

ABSTRACT

Attacks on natural language models are difficult to compare due to their different definitions of what constitutes a successful attack. We present a taxonomy of constraints to categorize these attacks. For each constraint, we present a real-world use case and a way to measure how well generated samples enforce the constraint. We then employ our framework to evaluate two state-of-the art attacks which fool models with synonym substitution. These attacks claim their adversarial perturbations preserve the semantics and syntactical correctness of the inputs, but our analysis shows these constraints are not strongly enforced. For a significant portion of these adversarial examples, a grammar checker detects an increase in errors. Additionally, human studies indicate that many of these adversarial examples diverge in semantic meaning from the input or do not appear to be human-written. Finally, we highlight the need for standardized evaluation of attacks that share constraints. Without shared evaluation metrics, it is up to researchers to set thresholds that determine the trade-off between attack quality and attack success. We recommend well-designed human studies to determine the best threshold to approximate human judgement.

1 INTRODUCTION

Advances in deep learning have led to impressive performance on many tasks, but models still make mistakes. Models are particularly vulernable to adversarial examples, inputs designed to fool models (Szegedy et al., 2014). Goodfellow et al. (2014) demonstrated that image classification models could be fooled by perturbations indistinguishable to humans.

Due to the importance of natural language processing (NLP) tasks, a large body of research has focused on applying the concept of adversarial examples to text, including (Alzantot et al., 2018; Jin et al., 2019; Kuleshov et al., 2018; Zhang et al., 2019; Ebrahimi et al., 2017; Gao et al., 2018; Li et al., 2018; Samanta & Mehta, 2017; Liang et al., 2017; Jia & Liang, 2017; Iyyer et al., 2018; Papernot et al., 2016a). The importance of tasks such as spam and plagiarism detection highlights the need for robust NLP models. However, there are fundamental differences between image and text data. Unlike images, two different sequences of text are never entirely indistinguishable. This raises the question: if indistinguishable perturbations aren't possible, what are adversarial examples in text?

We observe that each work from recent literature has a slightly different definition of what constitutes an adversarial example in natural language. Comparing the success rate of two attacks is meaningless if the attacks use different methods to evaluate the same constraints or define different constraints altogether. In this paper, we build on Gilmer et al. (2018) to introduce a taxonomy of constraints specific to adversarial examples in natural language. To the best of our knowledge, our work provides the first comprehensive framework for categorizing and evaluating attack constraints in natural language. We discuss use cases and propose standardized evaluation methods for each of these constraints.

We then apply our evaluation methods to the synonym-substitution based attacks of Jin et al. (2019) and Alzantot et al. (2018). These attacks claimed to preserve the syntax and semantics of the original sentence, while remaining non-suspicious to a human interpreter. However, we find that most of their adversarial examples contain additional grammatical errors, and human surveys reveal that many adversarial examples also change the meaning of the sentence and/or do not appear to be written by humans. These results call into question the ubiquity of synonym-based adversarial examples and emphasize the need for more careful evaluation of attack approaches.

Lastly, we discuss how previous works rely on arbitrary thresholds to determine the semantic similarity of two sentences. These thresholds can be tuned by the researcher to make their methods seem more successful with little penalty in quantitative metrics. Thus, we highlight the importance of standardized human evaluations to approximate the true threshold value. Any method that introduces a novel approach to measure semantic similarity should support their choice of threshold with defensible human studies.

The three main contributions of this paper are:

- We formally define and categorize constraints on adversarial examples in text, and introduce evaluation methods for each category.
- Using these categorizations and evaluation methods, we quantitatively disprove claims that state-of-the-art synonym-based substitutions preserve semantics and grammatical correctness.
- We show the sensitivity of attack success rate to changes in semantic similarity thresholds set by researchers. We assert that perturbations which claim semantic similarity should use standardized human evaluation studies with precise wording to determine an appropriate threshold.

2 BACKGROUND

Research has shown that current deep neural network models lack the ability to classify adversarial examples correctly (Szegedy et al., 2014). Two images that appear identical to a human but are different by a minuscule amount can be assigned grossly different classification scores by a model. These adversarial examples expose issues that arise from the black-box nature of deep learning predictions. Studying them is key to eventually building secure applications based on deep learning.

The search for adversarial examples can be framed as a simple optimization problem: maximizing the change in prediction while minimizing the change in input (Goodfellow et al., 2014; Szegedy et al., 2014; Papernot et al., 2016b; Carlini & Wagner, 2017). Past work in the image domain explored applying a small factor of the gradient to move an input in the "worst-case" direction. Zhao et al. (2017) applied a similar approach for text by mapping sentences into latent space. Since then, most approaches have been more heuristic-driven due to the difficulty of maintaining syntactic and semantic similarity otherwise. We categorize these attacks into four groups.

Attacks by Character Substitution: Recently, multiple studies proposed to attack natural language classification models by replacing words with deliberately misspelled words (Ebrahimi et al., 2017; Gao et al., 2018; Li et al., 2018). These studies used character replacements to change a word into one that the model doesn't recognize. Such replacements are designed to create character sequences that a human would easily correct into the intended words.

Character replacements, while effective in certain circumstances, can be detected with any spell-checking software. Besides, Pruthi et al. (2019) demonstrated a way to train a model that correctly classifies these misspelled inputs.

Attacks by Word Insertion or Removal: Liang et al. (2017) and Samanta & Mehta (2017) devised a way to determine the most important words in the input and then used heuristic-driven rules to generate perturbed inputs by adding or removing important words. Jia & Liang (2017) appended extra sentences to paragraphs that were inputs to reading comprehension tasks. These three studies each point out interesting flaws in natural language models, but do not produce adversarial examples that retain the semantics of the original input.

Attacks by Paraphrase: The bulk of recent work has defined adversarial examples in natural language as any sequences that both fool the model and share the *meaning* of the original input (Michel et al., 2019). Thus, if document $\mathbf{x}$ is of class y, then any paraphrase of $\mathbf{x}$ that is not of class y is a bona-fide adversarial example. Ribeiro et al. (2018) and Iyyer et al. (2018) proposed to use neural machine translation systems to paraphrase input text to create semantic-preserving perturbations. However, these systems have difficulty generating diverse, syntactically correct paraphrases. Their paraphrases often have bad grammar and overall have trouble fooling the model.

Attacks by Synonym Substitution: Due to the difficulty of generating paraphrases in general, Alzantot et al. (2018); Jin et al. (2019); Kuleshov et al. (2018); Papernot et al. (2016a) have developed easier ways to generate a subset of all paraphrases: by replacing a certain number of words from the input with synonyms.

Alzantot et al. (2018) proposed generating word-level swaps with a population-based optimization algorithm. The study aimed to generate semantically and syntactically similar adversarial examples

that fool well-trained sentiment analysis and textual entailment models. In order to select the best replacement word, the study computed the nearest neighbors of a selected word according to the distance in the counter-fitted embedding space. It also used the Google one billion words language model to filter out words that do not fit within the context.

Another recent system, TextFooler, was proposed by Jin et al. (2019) to attack three DNN models, including the powerful pre-trained BERT (Devlin et al., 2018), and the widely used convolutional and recurrent neural networks. The method identifies the most important words in the input sequence and replaces them one-by-one with synonyms until the model output changes. The study used counter-fitted word embeddings (Mrksic et al., 2016) to select the best replacement word and then utilized the Universal Sentence Encoder (Cer et al., 2018) to measure a similarity score between the perturbed and original inputs.

Jia et al. (2019) and Huang et al. (2019) used a technique called Interval Bound Propagation (IBP) to train models that withstand synonym attacks. IBP encourages models to make the same prediction on all sequences within a predefined neighborhood. In this case, it ensures that sentences that are a few synonym swaps away from each other have similar prediction scores. However, IBP is currently only feasible for simple models: feedforward networks, CNNs, and LSTMs with only a few layers. Perhaps more problematic, training with IBP causes a substantial drop in accuracy on the test set.

Barreno et al. (2006) define a broad framework to determine if a machine learning model is secure, but do not specify to adversarial examples on text. Gilmer et al. (2018) laid out a set of potential constraints for the attack space when generating adversarial examples, which are each useful in different real-world scenarios. Michel et al. (2019) defined a framework for evaluating attacks on machine translation models, focusing on meaning preservation constraints, but restricted their definitions to sequence-to-sequence models.

3 CONSTRAINTS ON ADVERSARIAL EXAMPLES IN NATURAL LANGUAGE

When constructing adversarial examples, it's important to define the space of inputs available to the attacker. A successful adversarial example is defined as one that fools the model, but other constraints are often defined to restrict the outputs that an attacker may produce.

For a given classifier F and a test input $\mathbf{x}$, we denote y_{gt} as the ground-truth output, $F(\mathbf{x})$ as the model output, and $C_0, C_1, ...C_n$ as a set of functional constraints, each of which represents whether an attack satisfies a certain constraint. An example $\mathbf{x}_{adv}$ is considered adversarial if:

$$F(\mathbf{x}_{adv}) \neq F(\mathbf{x}) \wedge C_i(\mathbf{x}, \mathbf{x}_{adv}) \quad \forall i \tag{1}$$

Adversarial examples aim to achieve two goals at the same time: (1) to induce misclassification by achieving $F(\mathbf{x}_{adv}) \neq y_{gt}$ (we focus on untargeted attacks), and (2) to restrict $\mathbf{x}_{adv}$. For instance, $C_0(\mathbf{x}, \mathbf{x}_{adv})$ requires morphology be preserved between $\mathbf{x}$ and $\mathbf{x}_{adv}$.

Unlike with images, text perturbations are never indistinguishable. The attacker must decide what makes two natural language inputs indistinguishable based on the scenario. This increases the importance of clearly defining constraints. Different sets of constraints are appropriate for different use cases. We extend on the categorization of attack spaces for adversarial examples introduced by (Gilmer et al., 2018) to introduce a taxonomy of constraints for natural language specifically, and discuss possible motivations for studying each constraint. We propose a method for evaluating whether adversarial examples meet each constraint.

We define three mutually exclusive categories of semantics-related constraints on attacks in natural language: altering an input sequence while changing as few characters as possible (morphologically-preserving), altering an input sequence while retaining its meaning (semantics-preserving), and crafting an input sequence which contains a specific message (semantics-constrained). We then define two constraints which may be used in addition to any semantic constraints: a requirement for adversarial examples to be grammatically correct, and the requirement to appear to have been written by a human (non-suspicious). Appendix A.1 categorizes a selection of prior work into these groups.

3.1 MORPHOLOGY-PRESERVING PERTURBATION

In some situations, the attacker is willing to change the semantics of the input as long as all changed sentences read the same to a human. These character substitutions may change the semantics of the sentence, as long as the reader can figure out the intended words from the context. If no defense is provided for morphological perturbations, an attacker can easily transmit a message by adding or a deleting a handful of characters (Gao et al., 2018).

A perturbation is morphology-preserving if satisfying $C(\mathbf{x}, \mathbf{x}_{adv}) := \{d(\mathbf{x}, \mathbf{x}_{adv}) \leq \epsilon\}$, where d measures the morphological distance between two strings. One common choice of d for morphological distance is $d_{edit}(x, x_{adv})$, the edit distance between $\mathbf{x}$ and $\mathbf{x}_{adv}$. Edit distance is the minimum number of operations (insertions, deletions, or substitutions) required to transform the string $\mathbf{x}_{adv}$ back into $\mathbf{x}$.

3.2 SEMANTICS-PRESERVING PERTURBATION

The attacker starts with an input sequence and is allowed to modify the input sequence however they want as long as the semantics is preserved. A use case for this class of adversarial examples is tricking plagiarism detection software. An attacker wants to preserve as much of the original document as possible but still avoid detection.

An adversarial example is semantics-preserving if a human labeler agrees that any changes between the input sequence $\mathbf{x}$ and the modified sequence $\mathbf{x}_{adv}$ do not alter the meaning of the sequence. We define $d_{sem}(\mathbf{x}, \mathbf{x}_{adv})$ as the score given by humans when asked to rate if the meaning is preserved on a Likert scale of 1-5, where 1 is "Strongly Agree" and 5 is "Strongly Disagree" (Likert, 1932; Jin et al., 2019). A perturbation is semantics-preserving if:

$$d_{sem}(\mathbf{x}, \mathbf{x}_{adv}) \leq \epsilon \tag{2}$$

We propose $\epsilon = 2$ as a general rule: on average, humans should either "agree" or "strongly agree" that $\mathbf{x}$ and $\mathbf{x}_{adv}$ have the same meaning.

Evaluation of semantics-preserving perturbations should ideally be done by humans, but machine evaluation of semantic similarity can be used as a proxy for human judgment, as explored by Michel et al. (2019). Automated metrics of meaning should always be supported by human studies.

3.3 SEMANTICS-CONSTRAINED INPUT

The attacker may generate any input sequence as long as it contains the semantic content that the attacker intends. This is distinct from the semantics-preserving case because there is no starting input sequence to perturb; the attacker can convey their message by any means possible. In these cases, the attacker can write anything, as long as it conveys specific meaning.

As discussed by Gilmer et al. (2018), one use case for semantics-constrained input is to fool a spam classifier. The attacker finds some way to relay their message that evades spam detection. The attacker may create any sequence as long as it contains the semantic content they desire. Examples with this constraint are more difficult to generate than semantics-preserving perturbations because the search space is much bigger and it is challenging to check that the model produces incorrect output without human review.

Evaluation of semantics-constrained input is more difficult than semantics-preserving perturbations, as there is no starting point to compare to. There will never be automated way to tell if an email is spam with 100% accuracy. Whether an input is semantics-constrained must be determined by a human judge.

3.4 SYNTACTIC CONSTRAINT

Under this constraint, the attacker is constrained to inputs that are grammatically valid. Perturbations that introduce grammar errors often aren't semantics-preserving. However, in many cases, a human reader will recognize that $\mathbf{x}$ and $\mathbf{x}_{adv}$ have the same meaning even if $\mathbf{x}_{adv}$ introduced additional syntactic errors. There may be use cases for both semantics-preserving perturbations and semantics-constrained inputs with and without syntactic constraints:

- **Semantics-preserving perturbation with syntactic constraint**: Consider a student who wishes to alter a copied assignment to evade plagiarism detection. They must perturb the assignment while preserving the meaning, but introducing grammatical errors would get them a bad grade.
- **Semantics-preserving perturbation without syntactic constraint**: An attacker who wishes to illegally distribute a PDF online must retain the content in the PDF, but readers of the PDF are indifferent to whether grammar errors are introduced as long as the meaning is unchanged. Similar to the case of illegal video streaming discussed in Gilmer et al. (2018), the consumers of the PDF may be thought of as colluding with the attacker.
- **Semantics-constrained input with syntactic constraint**: An adversary wishing to evade a fake news classifier must generate an article with correct grammar.
- **Semantics-constrained input without syntactic constraint**: Someone wishing to post offensive content on social media may be willing to purposely misspell words or insert grammatical errors in order to get their content onto a platform.

Evaluation of whether adversarial examples follow the grammatical constraint requires a way of measuring the amount of grammar errors in a document. Define $G(\mathbf{s})$ as the number of syntactic errors in an input sequence $\mathbf{s}$. When applying a semantics-preserving perturbation from $\mathbf{x}$ to $\mathbf{x}_{adv}$, the grammatical constraint may be represented as

$$G(\mathbf{x}_{adv}) - G(\mathbf{x}) \leq \epsilon \tag{3}$$

where ϵ represents the amount of additional grammatical errors the adversary is willing to introduce. When generating input from scratch, the constraint becomes simply: $G(\mathbf{x}_{adv}) \leq \epsilon$, where ϵ is the amount of total grammatical errors the adversary is willing to create. Methods which claim to introduce no grammatical errors imply $\epsilon = 0$. Evaluation of $G(\mathbf{x})$ and $G(\mathbf{x}_{adv})$ may be done by humans, but pragmatically it is more convenient and consistent to use an automatic grammar checker such as LanguageTool (Naber, 2003).

3.5 Non-suspicious constraint

The non-suspicious constraint specifies that the adversarial example must appear to be human-written to other humans. It has some overlap with the syntactic constraint, but they do not always go together. Sentences without grammar errors may still not seem human-written. Conversely, grammatically incorrect sentences could always have been written by a human. An example use case of an attack with the non-suspicious constraint is the plagiarism example discussed above: the student would not want to make modifications to their assignment such that the teacher would be able to tell the assignment was not human-written. A case where the non-suspicious constraint does not apply is the illegal PDF as mentioned earlier, as the consumers of the PDF do not care if it has been altered as long as meaning is preserved.

Evaluation of the non-suspicious constraint must be done by humans. We propose a method in which humans are shown a shuffled mix of real and adversarial sequences and must guess whether each one is real or computer-altered, or computer-generated in the case of semantics-constrained input. Define the portion of people who correctly identify an adversarial example $\mathbf{x}_{adv}$ as computer-altered as $R(\mathbf{x}_{adv})$ An adversarial example is considered to meet the non-suspicious constraint if $R(\mathbf{x}_{adv}) < \epsilon$, where $0 < \epsilon \leq 1$. A smaller value of ϵ enforces more rigorous standards on what must appear to be written by a human.

4 Experiments

We now apply the evaluation framework discussed in Section 3 to evaluate the effectiveness of the attack techniques from Alzantot et al. (2018) and Jin et al. (2019). We chose these these works because:

- They claim to find semantics-preserving perturbations which adhere to the syntactic constraint, and imply adherence to the non-suspicious constraint. However, our inspection of the adversarial perturbations revealed that many introduced syntax errors or did not preserve semantics.
- They report high attack success rates[1].
- These methods cover two of the most effective models for text classification: LSTM and BERT.

To generate examples for evaluation, we attacked BERT using Jin et al. (2019)'s method and attacked a WordLSTM using Alzantot et al. (2018)'s method. We only considered classification tasks as our focus is on evaluating the adversarial examples. We evaluate both methods on the IMDB[2] and Yelp polarity document-level sentiment classification datasets and Jin et al. (2019)'s method on the MR sentence-level sentiment classification dataset (Pang & Lee, 2005; Zhang et al., 2015). We ran experiments to evaluate whether the generated examples met three constraints: syntax, semantics-preservation and non-suspicion.

4.1 Evaluation of Grammatical Correctness

We evaluate fulfillment of the syntactic constraint using a grammar checker. These programs are easy to use, free, and cover a wide range of grammatical rules over many languages. We chose to use LanguageTool, an open-source proofreading tool (Naber, 2003). LanguageTool ships with thousands of human-curated rules for the English language and provides a downloadable server interface for analyzing sentences.

[1] We use "attack success rate" to mean the percentage of the time that an attack can find an adversarial example by perturbing a given input. "After-attack accuracy" or "accuracy after attack" is the accuracy the model achieves on a set of adversarial examples.

[2] https://datasets.imdbws.com/

We ran each of the generated $(\mathbf{x}, \mathbf{x}_{adv})$ pairs through LanguageTool and compared the amount of detected errors to determine if the syntactic constraint as defined in Equation 3 was fulfilled. LanguageTool detected more grammatical errors in $\mathbf{x}_{adv}$ for 52.1% of pairs across the five datasets. There is a clear linear relationship between the number of words changed and the number of grammatical errors induced (Figure 1). Clearly, the majority of generated examples don't fulfill the syntactic constraint.

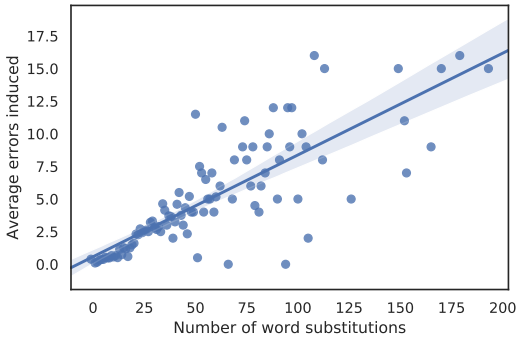

Figure 1: Relationship between number of substitutions and grammatical errors induced.

As shown in Table 1, grammatical errors were introduced more often in adversarial examples generated by Jin et al. (2019) than Alzantot et al. (2018). While Jin et al. (2019) did not filter substitutions by syntax, Alzantot et al. (2018) judged the grammaticality of each replacement candidate with a language model.

	(Jin et al., 2019)			(Alzantot et al., 2018)	
	IMDB	Yelp	MR	IMDB	Yelp
Average score	61.8%	71.6%	35.5%	42.7%	43.8%

Table 1: Percentage of $\mathbf{x}_{adv}$ with more errors than $\mathbf{x}$ by attack method and dataset.

A savvy defender who wants to build a system robust against these types of substitution attacks may notice that adversarial examples contain strange phrasings and unnatural errors that a human would almost never make. LanguageTool detected errors in several categories that appeared far often in adversarial samples than in the original, human-generated content.

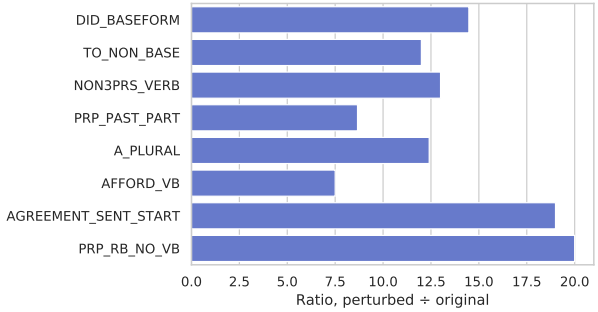

Figure 2: Errors that appear 5x more often in perturbed sample.

Figure 2 shows the ratio of appearance of grammatical errors in perturbed vs. original samples. Six different classes of mistakes appear over 10 times more frequently in the adversarial examples than original inputs. A description of each type of error is in the appendix.

4.2 Evaluation of Preservation of Semantics

As discussed in Section 3.2, tools for automatically evaluating semantic similarity exist, but human judgement is essential to confirm such automatic evaluations. While Jin et al. (2019) and Alzantot et al. (2018) both include human evaluation, these studies include a single scale on which users rated the similarity of $\mathbf{x}$ and $\mathbf{x}_{adv}$ to be relatively high. We believe that this phrasing did not generate an accurate measure of semantic similarity, as users may have considered two sentences with clearly different meanings to be "similar" due to their small morphological distance. Instead, we ask users to determine whether the changes between $\mathbf{x}$ and $\mathbf{x}_{adv}$ preserve the meaning of the passage.

We set up our own survey to evaluate human judgement on the adversarial examples from Jin et al. (2019) and Alzantot et al. (2018). We enlisted workers from Amazon Mechanical Turk to label 3907 data points across the five datasets. Our survey randomized the order of $\mathbf{x}$ and $\mathbf{x}_{adv}$ and displayed them side-by-side. To evaluate whether the perturbation was semantics-preserving as defined in Equation 2, we asked users to rate whether they agreed that the changes between the two passages preserved the meaning of the passage on a scale of 1 (Strongly Agree) to 5 (Strongly Disagree).

	(Jin et al., 2019)			(Alzantot et al., 2018)	
	IMDB	Yelp	MR	IMDB	Yelp
Average score	2.68	2.86	2.82	3.41	3.62

Table 2: Likert scale response: Does changing from $\mathbf{x}$ to $\mathbf{x}_{adv}$ preserve meaning?

Average scores as shown in Figure 2 vary by dataset, but users' answers generally generally average out to around 3 ("Not Sure"). Clarifying the survey question makes it clear that many of the examples are not semantics-preserving.

Additionally, we note that due to acquiescence bias, labelers are generally more likely to agree than disagree when presented with a statement (Jackson & Messick, 1961) [3]. To test this effect, we negated the question: instead of asking whether the changes from $\mathbf{x}$ to $\mathbf{x}_{adv}$ "preserved" the meaning of $\mathbf{x}$, we asked whether the differences "changed" the meaning.

	(Jin et al., 2019)			(Alzantot et al., 2018)	
	IMDB	Yelp	MR	IMDB	Yelp
Average score	2.35	2.26	2.39	2.12	2.17

Table 3: Likert scale response: Does changing from $\mathbf{x}$ to $\mathbf{x}_{adv}$ change meaning?

Table 3 shows that averages are now closest to "2 - Agree" rather than "3 - Not sure". Thus, inverting the question resulted in generated examples being rated as less semantics-preserving. Future human evaluation studies should use similar practices to minimize the effects of this bias.

4.3 Evaluation of Non-suspicion

The final claim we investigated was whether adversarial examples generated by these two attack techniques were truly non-suspicious. We presented examples one by one to humans and asked if they were real or computer-altered as proposed in Section 3.5. A method that consistently generated non-suspicious adversarial examples would produce inputs would appear to be human-written. In this case, human labelers would average an accuracy of 50%. If humans could consistently discern perturbed inputs from real inputs, they would achieve 100% on this task.

As this is a time-consuming task for long documents, we only evaluated adversarial examples generated by (Jin et al., 2019)'s method on the sentence-level MR dataset. We sampled 100 ($\mathbf{x}$, $\mathbf{x}_{adv}$) pairs then used Mechanical Turk to get 10 guesses for each example. Humans achieved 69.2% accuracy on this task. Table 4 presents the confusion matrix of results from the survey. Interestingly, workers guessed that the examples were real 62.2% of the time, but when they guessed that examples were computer-altered they were right 75.4% of the time. Thus while some perturbed examples are non-suspicious, there are some which workers identify with high precision.

		Guessed Label	
		Real	Computer-altered
True Label	Original	814	431
	Perturbed	186	570

Table 4: Confusion matrix for humans guessing if perturbed examples are computer-altered

5 To Need Standardized Metrics Supported By Human Evaluation

A method that detects or generates semantics-preserving adversarial examples relies on a distance metric d to measure how well two sentences align in meaning. Two inputs (x_1, x_2) are paraphrases

[3] For more discussion, see Section A.2

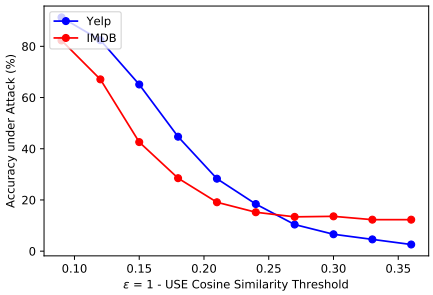

Figure 3: ϵ = 1 - USE Similarity Threshold vs. Accuracy under attack

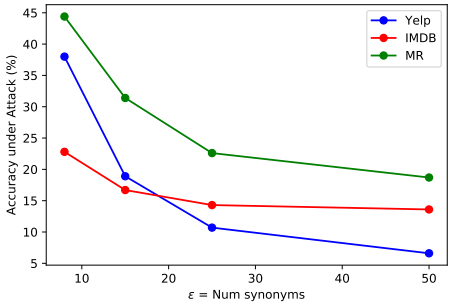

Figure 4: ϵ = Num synonyms vs. Accuracy under attack

(i.e. semantic-preserving) if $d(x_1, x_2) < \epsilon$ for some fixed threshold ϵ. Often, a single attack relies on multiple metrics for d, each with its own threshold ϵ.

Sometimes, prior work has chosen the same metric d but different thresholds for ϵ. Synonym substitution attacks provide one example. These attacks consider two sentences $(\mathbf{x}, \mathbf{x}_{adv})$ that differ by a single word $(w \in \mathbf{x}, w_{adv} \in \mathbf{x}_{adv})$ to have the same meaning if w_{adv} is one of the closest ϵ neighbors in the counter-fitted embedding space. In other words, ϵ indicates how many synonyms they allowed. At least three different such attacks rely on this metric: Alzantot et al. (2018) uses $\epsilon = 8$, Kuleshov et al. (2018) uses $\epsilon = 15$, and Jin et al. (2019) uses $\epsilon = 50$. Without an agreed-upon standard value, choice of ϵ is at the researcher's discretion. This is problematic because there is a direct correlation between the value chosen for ϵ and the success of an attack.

Jin et al. (2019) used an additional distance metric d defined as the cosine similarity between embeddings encoded by the Universal Sentence Encoder (USE) in order to determine if a synonym swap preserves semantic similarity (Cer et al., 2018). Figure 3 shows accuracy under BERT under attack by Jin et al. (2019)'s method as the maximum allowed cosine similarity between two sentences' USE embeddings increases [4] . As ϵ becomes more strict, the attack becomes less successful. Figure 4 plots the accuracy under attack as the number of synonyms considered for each substitution increases. An attack that is more lenient with its standard for what constitutes a synonym is more successful.

For any distance metric, there is no perfect value of ϵ. Whether two sentences are truly paraphrases is often subjective. We cannot expect any automated evaluation tool to perfectly separate paraphrases from non-paraphrases. However, there will be a range of values for ϵ that correlate the most closely with the judgement of humans. Without using standardized human studies like the one in Section 4.2 to verify the linguistic standard provided by the choice of ϵ, generated adversarial examples may not be useful at all.

The correct range of values of ϵ will vary by metric. Every study that presents a new way to measure semantic similarity should perform a human study to approximate the proper value for ϵ. With consistent values for ϵ, researchers will be able to easily compare methods that use the same metrics.

6 Conclusion

We introduced a framework for evaluating fulfillment of attack constraints in natural language. Applying this framework to synonym substitution attacks raised concerns about the semantic preservation, syntactic accuracy, and conspicuity of the adversarial examples they generate.

Future work may expand our hierarchy to categorize and evaluate different attack constraints in natural language. Standardized terminology and evaluation metrics will make it easier for defenders to determine which attacks they must protect themselves from–and how. It remains to be seen how robust BERT is when subject to synonym attacks which rigorously preserve semantics and syntax. It is up to future research to determine how prevalent adversarial examples are throughout the broader space of paraphrases.

[4] The MR dataset is excluded due to no USE similarity restrictions being enforced on inputs of less than 15 words.

REFERENCES

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*, 2018.

Marco Barreno, Blaine Nelson, Russell Sears, Anthony D. Joseph, and J. D. Tygar. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, ASIACCS '06, pp. 16–25, New York, NY, USA, 2006. ACM. ISBN 1-59593-272-0. doi: 10.1145/1128817.1128824. URL http://doi.acm.org/10.1145/1128817.1128824.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pp. 39–57. IEEE, 2017.

Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. *ArXiv*, abs/1803.11175, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2018.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*, 2017.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *IEEE Security and Privacy Workshops (SPW)*, 2018.

Justin Gilmer, Ryan P. Adams, Ian J. Goodfellow, David Andersen, and George E. Dahl. Motivating the rules of the game for adversarial example research. *CoRR*, abs/1807.06732, 2018. URL http://arxiv.org/abs/1807.06732.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. Achieving verified robustness to symbol substitutions via interval bound propagation. *ArXiv*, abs/1909.01492, 2019.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. *CoRR*, abs/1804.06059, 2018. URL http://arxiv.org/abs/1804.06059.

Douglas N. Jackson and Samuel Messick. Acquiescence and desirability as response determinants on the mmpi. *Educational and Psychological Measurement*, 21(4):771–790, 1961. doi: 10.1177/001316446102100402. URL https://doi.org/10.1177/001316446102100402.

Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*, 2017.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. Certified robustness to adversarial word substitutions, 2019.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *arXiv e-prints*, art. arXiv:1907.11932, Jul 2019.

Volodymyr Kuleshov, Shantanu Thakoor, Tingfung Lau, and Stefano Ermon. Adversarial examples for natural language classification problems. 2018.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*, 2018.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. Deep text classification can be fooled. *arXiv preprint arXiv:1704.08006*, 2017.

R. Likert. *A Technique for the Measurement of Attitudes*. Number nos. 136-165 in A Technique for the Measurement of Attitudes. publisher not identified, 1932. URL https://books.google.com/books?id=9rotAAAAYAAJ.

Daniel Lowd and Christopher Meek. Good word attacks on statistical spam filters. In *CEAS*, volume 2005, 2005.

Paul Michel, Xian Li, Graham Neubig, and Juan Miguel Pino. On evaluation of adversarial perturbations for sequence-to-sequence models. *CoRR*, abs/1903.06620, 2019. URL http://arxiv.org/abs/1903.06620.

Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina Maria Rojas-Barahona, Pei hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. Counter-fitting word vectors to linguistic constraints. In *HLT-NAACL*, 2016.

Daniel Naber. A rule-based style and grammar checker. 01 2003.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp. 115–124, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219855. URL https://www.aclweb.org/anthology/P05-1015.

Nianolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. Crafting adversarial input sequences for recurrent neural networks. In *Military Communications Conference, MILCOM 2016-2016 IEEE*, pp. 49–54. IEEE, 2016a.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2016b.

Danish Pruthi, Bhuwan Dhingra, and Zachary C Lipton. Combating adversarial misspellings with robust word recognition. *arXiv preprint arXiv:1905.11268*, 2019.

Marco Tulio Ribeiro, Sameer Singh, and Carlos 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)*, pp. 856–865, 2018.

Suranjana Samanta and Sameep Mehta. Towards crafting text adversarial samples. *arXiv preprint arXiv:1707.02812*, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

Wei Emma Zhang, Quan Z. Sheng, and Ahoud Abdulrahmn F. Alhazmi. Generating textual adversarial examples for deep learning models: A survey. *CoRR*, abs/1901.06796, 2019. URL http://arxiv.org/abs/1901.06796.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pp. 649–657, 2015.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*, 2017.

A APPENDIX

A.1 CATEGORIZATION OF NLP ATTACKS

Past research in attacks on textual models has provided many disparate sets of constraints for adversarial examples. Here, we select a small slice of the past work and categorize it based on the taxonomy outlined in Section 3.

Attack	Constraint on Meaning	Syntactic Constraint	Non-suspicious
Synonym Substitution. (Alzantot et al., 2018; Kuleshov et al., 2018; Jin et al., 2019)	semantics-preserving perturbation	✓	✓
Character Substitution. (Ebrahimi et al., 2017; Gao et al., 2018; Li et al., 2018)	morphology-preserving perturbation	✗	✓
Word Insertion or Removal. (Liang et al., 2017; Samanta & Mehta, 2017)	semantics-preserving perturbation	✗	✗
General Paraphrase. (Zhao et al., 2017; Ribeiro et al., 2018; Iyyer et al., 2018)	semantics-preserving perturbation	✓	✓
Good Word Attack. (Lowd & Meek, 2005)	semantics-preserving perturbation	✗	✗
Sentence Insertion. (Jia & Liang, 2017)	semantics-preserving perturbation	✓	✗

A.2 DETAILS ABOUT HUMAN STUDIES.

Our experiments relied on labor crowd-sourced from Amazon Mechanical Turk to provide labels for two tasks. We used five datasets: MIT and Yelp datasets from (Alzantot et al., 2018) and MIT, Yelp, and Movie Review datasets from (Jin et al., 2019). We added a limitation so that only workers with "Masters" status on Mechanical Turk could complete our tasks. We estimated that each task would take approximately 15 seconds to complete, so we paid workers $\$.05$ per label to ensure a fair wage of $\$12$ per hour. For both tasks, we allotted workers 3 minutes per assignment.

Task 1: Real vs. fake? In Section 4.3, we present results from our Mechanical Turk survey where we asked users to guess if a sample of text was real or Computer-altered. We restricted this task to a single dataset, Movie Review. We chose Movie Review because it had an average sample length of 20 words, much shorter than Yelp or IMDB. We made this restriction because of the time-consuming nature of classifying long samples as Real or Fake.

Task 2: Semantic Similarity. In Section 4.2, we present results from two Mechanical Turk questionnaires to judge semantic similarity or dissimilarity. For each task, we show x and x_{adv}, in a random order. We added a custom script to highlight the different characters between the two sequences. For both tasks, we provided the following description: "Compare two short pieces of English text and determine if they mean different things or the same." We then prompted labelers: "Changing from one of these sentences to the other X the meaning," where X was "changes" or "preserves".

Phrasing matters. Mechanical Turk comes with a set of pre-designed questionnaire interfaces. These include one titled "Semantic Similarity" which asks users to rate a pair of sentences on a scale from "Not Similar At All" to "Highly Similar." Examples generated by synonym attacks benefit from this question formulation because humans tend to rate two sentences that share many words as "Similar" due to their small morphological distance, even if they have different meanings.

Cognitive biases in the Likert scale. It was interesting to compare results from the surveys within Task 2. Since all we did was negate the question, the results should have been inverses: i.e., every label that said "Agree" for the first formulation of the question should have said "Disagree" for the second, etc. However, we found that labelers, at least on the Mechanical Turk, tend to select "Agree" regardless of the question. When asked if the change from x to x_{adv} "changes" meaning, 68.3% of labelers selected "Agree". Surprisingly, when asked if the change from x to x_{adv} "preserves" meaning, 43.6% of labelers *still* answered with "Agree"! Overall, 55.9% of our 7,816 responses, from two opposite surveys, were simply "Agree". We believe that this is an instance of a phenomenon called acquiescence bias, where respondents to a survey gravitate towards answers that are generally positive (Jackson & Messick, 1961).

Notes for future surveys. In the future, we would try to filter out bad labels by mixing a small number of ground-truth "easy" data points into our dataset and rejecting the work of labelers who performed poorly on this set.

A.3 LANGUAGETOOL ERROR TYPES

LanguageTool presents helpful error message along with each detected grammatical rule violation. This table displays sample error messages alongside the error codes from Figure 2. For example, code `PRP_RB_NO_VB` appears 20 times more frequently in adversarial samples than it does in real, human-crafted input. This specific error often appears when the attack substitutes a verb with a noun.

	Error ID	Ratio	Example Error Message
1	`PRP_RB_NO_VB`	20.0	Are you missing a verb?
2	`AGREEMENT_SENT_START`	19.0	You should probably use 'have', 'haven'.
3	`DID_BASEFORM`	14.5	The verb 'should' requires the base form of the verb: 'constitute'
4	`NON3PRS_VERB`	13.0	The pronoun 'we' must be used with a non-third-person form of a verb: 'summon', 'summons'
5	`A_PLURAL`	12.4	Don't use indefinite articles with plural words. Did you mean 'a circumstance' or simply 'circumstances'?
6	`TO_NON_BASE`	12.0	The verb after "to" should be in the base form: 'constitute'.
7	`PRP_PAST_PART`	8.7	Possible grammatical error. You used a past participle without using any required verb ('be' or 'have'). Did you mean 'was', 'were'?
8	`AFFORD_VB`	7.5	This verb is used with the infinitive: 'to better', 'to well'