ValCAT: Generating Variable-Length Contextualized Adversarial Transformations using Encoder-Decoder

Anonymous ACL submission

Abstract

Adversarial samples are helpful to explore vulnerabilities in neural network models, improve model robustness, and explain their working mechanism. However, the adversarial texts generated by existing word substitution-based methods are trapped in a one-to-one attack pattern, which is inflexible and cramped. In this paper, we propose **ValCAT**, a black-box attack framework that misleads the language model by applying variable-length contextualized transformations to the original text. Experiments show that our method outperforms state-of-the-art methods on attacking several classification tasks and inference tasks. More comprehensive human evaluations demonstrate that ValCAT has a significant advantage in ensuring the fluency of the adversarial samples and achieves better semantic consistency. We release our code at https://github.com/linerxliner/ValCAT.

1 Introduction

Deep learning is successfully applied in a variety of fields, while previous works have found that neural network models are vulnerable to adversarial samples (Goodfellow et al., 2014; Kurakin et al., 2016). Adversarial samples are constructed with small perturbations to original inputs to fool these models with incorrect decisions while being imperceptible to humans. Therefore, exploring adversarial samples is essential to improve the performance of neural network models with higher reliability and robustness. However, compared to the long-studied image domain, generating adversarial samples on text is more difficult because texts are discrete, where small changes can alter the original meaning and make it unnatural (Xu et al., 2020; Zhang et al., 2020).

Word substitution-based attack methods have received much attention in the recent past. Several previous works explore the substitution based only on the properties of individual words, with the help of inflectional morphology (Tan et al., 2020), counter-fitting word vectors (Jin et al., 2020), and sememe (Zang et al., 2020), etc. The encoder language models, like BERT, give us new insights of considering context information for the substitution candidate generation (Li et al., 2020; Garg and Ramakrishnan, 2020; Li et al., 2021). However, all these works focus on single-word substitution like examples shown in Table 1. This one-to-one attack pattern limits the perturbation forms and overlooks the interactions between words. However, the possibility of adding perturbations to larger semantic units has not been discussed. For this sake, two research questions are raised: 1) *How to take into account interactions between words in vulnerable-position discovery?* 2) *How to perturb vulnerable*

AG News (Business)	Lucent milestone: A profit Lucent Technologies yesterday posted higher fiscal fourth-quarter earnings, helping lift the telecommunications equipment maker to its first profitable year since 2000.
BERT-Attack (Sci/Tech)	Lucent node: A revenue Lucent tech yesterday reported higher revenue fourth-quarter benefits, which lift the multimedia equipment maker to its first business year year 2000.
ValCAT (Sci/Tech)	Lucent milestone: A profit Lucent [IT] Technologies [recently reported] higher fiscal fourth-quarter earnings, helping lift the telecommunications equipment maker to its first profitable year since 2000.
MNLI (Neutral)	*Premise*: He caught a grip on himself, fighting the fantasies of his mind, and took another breath of air. *Hypothesis*: The air tasted like molten metal - the taste of blood.
BERT-Attack (Contradiction)	*Hypothesis*: The air tasted through boiling armor - the taste of betrayal.
ValCAT (Contradiction)	*Hypothesis*: The air tasted [nothing like air] - the taste of blood.

Table 1: Examples of adversarial texts generated by ValCAT and BERT-Attack. The first one is the original text, followed by adversarial texts generated by ValCAT and BERT-Attack.

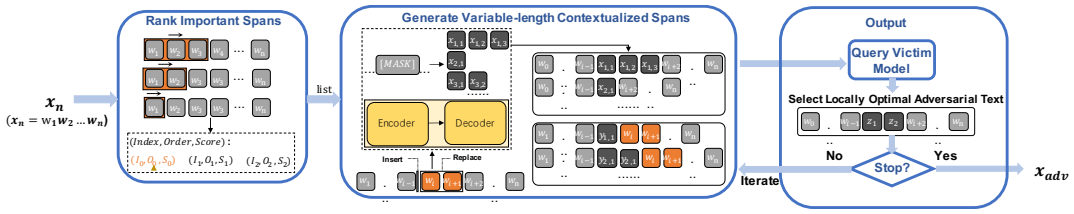

Figure 1: Overview of ValCAT's workflow.

positions with variable-length contextualized substitution?

We propose ValCAT, which generates high-quality adversarial texts by applying variable-length contextualized transformations to the original text. Specifically, given a benign text, we enumerate all possible spans by traversing the text with sliding windows of different sizes and evaluate their importance. Based on the importance of each span, we propose two operations, *Replace* and *Insert* to generate adversarial candidates by an encoder-decoder language model. The encoder can recover a mask token using a single word while the decoder can only predict words after a prompt. Therefore, joint use of encoder and decoder enables ValCAT to generate variable-length contextualized candidates at the arbitrary vulnerable position.

Furthermore, we evaluate ValCAT by attacking fine-tuned BERT on several classification tasks and inference tasks. Experimental results show that it outperforms other baseline methods in attack success rate, fluency and similarity. In particular, we observe that the variable-length property can significantly reduce perplexity, which is less than 50% compared to the best baseline. Although each multi-word transformation perturbs more words than one-word transformation, it requires fewer transformations toward success. This in general results in a tolerable perturbation rate, which is even lower on some of the inference datasets. A comprehensive human evaluation further verifies that ValCAT has significant advantages in both fluency&grammaticality and semantic similarity. The main contributions of this paper are summarized as follows:

- We propose ValCAT, the first variable-length contextualized adversarial attack against NLP models.
- Our work is the first to propose *Sliding Window* for vulnerable-position discovery and variable-length adversarial candidate generation, which fully exploits the advantage of encoder-decoder models.
- Automatic evaluations and comprehensive human evaluation demonstrates the superior quality of our generated adversarial samples.

2 ValCAT

To further improve the attack effectiveness and simultaneously improve the fluency and semantic similarity of the adversarial samples, we propose **ValCAT**, which can generate high-quality adversarial text by applying variable-length contextualized transformations with a joint encoder-decoder framework.

Problem Formalization Given a victim model $F : X \rightarrow Y$ and a text $x = w_1 w_2 ... w_{n-1} w_n$ that can be correctly classified by F, the attack goal is to generate an adversarial text $\tilde{x}$, which can flip the model prediction, i.e. $F(\tilde{x}) \neq F(x)$. A continuous word sequence with length n can be denoted as a span s_n. In the soft-label black-box setting, the attacker only has access to the logit output $P(y|x)$. The architecture, parameters and configurations of F are unknown to attacker. To achieve human imperceptibility, the attacker should minimize textual perturbations and maintain semantic consistency.

ValCAT ValCAT performs the attack in a sequential manner, as the workflow in Figure 1. To locate the most-vulnerable positions for the perturbations in each iteration, ValCAT first rank the importance of s_n in x with a sliding window of length 1 to MAX, as line 2-6 in Algorithm 1. Based on the sorted ranking list R, ValCAT uses encoder-decoder language model to generate variable-length contextualized spans, as line 8-13. Using two perturbations in line 9, REPLACE and INSERT, we obtain a candidate set T. If some of the candidates can mislead the victim model, ValCAT declares the one with the highest cosine similarity with the original text as the final successful result, as line 11. However, if no candidate successes at this iteration, we select the one with the highest negative impact to the victim model as the basic text for the next iteration, as line 13. Note that, if a span in the ranked list has been selected as the target span, the subsequent

spans which are overlapped with the target span will be removed from the ranked list to avoid multiple modifications on a token. The sequential attack ends when successful attack occurs, or when the upper limit of the perturbation constraints (See Section 3.1) are reached. The latter case is considered as a final failure. Below we elaborate on the two stages of the attack in Section 2.1 and Section 2.2, in detail.

2.1 Important Span Ranking

Echoing the observation to prior works (Niven and Kao, 2019; Jin et al., 2020), only some key words act as most-vulnerable positions for the victim model F. Perturbations over these words can be most beneficial in crafting adversarial texts. Considering the interactions between words, ValCAT performs transformations on several important spans instead of single words for each perturbation.

Given a text x, we evaluate the importance of a span s within x according to how removing the span can impact the model prediction, in the black-box setting. Let $\tilde{x}_s$ denote the text of removing s from x. Formally, we define the importance of s with respect to x as:

$$I_x(s) = \begin{cases} d_y(x, \tilde{x}_s), & if \quad y = \tilde{y} \\ d_y(x, \tilde{x}_s) + d_{\tilde{y}}(\tilde{x}_s, x), & if \quad y \neq \tilde{y} \end{cases}$$

where y and $\tilde{y}$ are the predictions of x and $\tilde{x}_s$, respectively, and $d_y(x, \tilde{x}_s) = P(y|x) - P(y|\tilde{x}_s)$ is the difference of the probabilities that x and $\tilde{x}$ are classified as y.

To compare spans of different lengths, we propose *Sliding Windows* to measure the importance of variable-length spans. Specifically, we apply multiple sliding windows of the corresponding size, which traverse the text from left to right. Each span bounded by a sliding window is sequentially deleted from the original text for its importance calculation. Finally, we obtain a set of triples each consisting of the length, the start position, and the importance score of a span. We rank the triples according to the importance score in descending order.

2.2 Variable-Length Contextualized Transformations

Based on the triple ranking list, ValCAT performs sequential perturbations, where each step a target span in the original text is replaced by or inserted with a set of adversarial spans generated by an encoder-decoder language model. The variable-length of the adversarial spans renders the language model enough space to produce more contextually appropriate candidates to improve fluency. Meanwhile, our variable-length method expands the perturbation forms, for it supports multi-word transformations while is compatible with traditional one-to-one transformations. Compared with previous methods, this further improves the attack success rate under the same perturbation constraints. Below we elaborate on the details of the adversarial text generation.

Algorithm 1: VALCAT

Input: Victim Model F; Text x; Label y; Maximum size of slide window M
Output: Adversarial sample

$R \leftarrow \emptyset$; $t \leftarrow x$
for w = 1 *to* M **do**
  for i = 1 *to* LEN$(x) - w + 1$ **do**
    $s \leftarrow x_{i,\ldots,i+w-1}$
    Calculate span importance $I_x(s)$
    $R \leftarrow R \cup \langle i, w, I_x(s) \rangle$
Sort R by $I_x(s)$ in descending order
for *(i, w, _) in* R **do**
  $T \leftarrow$ REPLACE$(t, i, w) \cup$ INSERT(t, i)
  if $\exists \tilde{x} \in T$ s. t. $F(\tilde{x}) \neq y$ **then**
    return $\underset{\tilde{x} \in T, F(\tilde{x}) \neq y}{\text{argmax}}$ SIMILAR$(\tilde{x}, x)$
  else
    $t \leftarrow \arg\max_{\tilde{x} \in T} |F(\tilde{x}) - y|$
return NULL

Adversarial Span Generation To generate candidates of each adversarial span, ValCAT applies an encoder-decoder language model to fill the mask token with a list of variable-length predictions. First, the encoder model confer the capability of predicting the masked tokens, which is trained with the masked language modeling (MLM) objective. However, the encoder model fills one mask token with only one suitable substitute rather than multiple words. The ability of the decoder could fill the gap of the single word, since the decoder is trained with a causal language modeling (CLM) which can predict the sequence after a prompt. But, the decoder can only generate sequences at the end of the text. Hence, ValCAT combined these two

models with their complementary advantages, with the predictive capability at arbitrary positions of encoder and variable-length generation of a decoder. With the candidate spans for target span, ValCAT performs two kinds of perturbation, *Replace* and *Insert* to generate the adversarial candidates.

Replace The *Replace* operation substitutes the target span $s_m = w_i...w_{i+m-1}$ with another $\tilde{s}$. For example, the target "*awesome*" in the text "This place is awesome." could be replaced by the adversarial span "*pretty good*". Specifically, we first replace a mask token $[\text{mask}]$ to s_m:

$$\tilde{x}^{[i:i+m]} = w_1...w_{i-1}[\text{mask}]w_{i+m}...w_n,$$

and generate a set of variable-length adversarial spans $\mathcal{Z}$ to fill the mask. The adversarial text is denoted as:

$$\tilde{x}_z^{[i:i+m]} = w_1...w_{i-1}zw_{i+m+1}...w_n,$$

where $z \in \mathcal{Z}$ is a contextualized span.

Since the language model is blind to the information of the target span, some of the generated adversarial spans may deviate from the original meaning to a large extent. To avoid this situation, we only keep the adversarial spans with a high degree of semantic similarity to the original span. Specifically, we use Universal Sentence Encoder (Cer et al., 2018) to restrict their cosine semantic similarity. We also impose a limit on the word perturbation rate (See Section 3.1). To prevent the text from being too long, we constrain the adversarial spans to be at most two words longer than the target span.

Insert. The *Insert* operation inserts a new span $\tilde{s}$ in front of the target span s_m. For example, "I like this *quite interesting* movie.". Similar to the Replace operation, it inserts a mask token in front of the target span:

$$\tilde{x}^{i} = w_1...w_{i-1}[\text{mask}][w_i...w_{i+m-1}]w_{i+m}...w_n,$$

and corresponds with the adversarial text $\tilde{x}_z^i = w_1...w_{i-1}\ z\ w_i...w_n$. The *Insert* perturbation also follows the same perturbation constraints, mentioned in Section 3.1.

3 Experiments

In this section, we evaluate ValCAT on two NLP tasks, text classification and natural language inference. To demonstrate the effectiveness of ValCAT in terms of fluency&grammaticality and semantic similarity, following Li et al. 2021, we design and conduct a comprehensive human evaluation.

3.1 Implementation

Victim model In this work, we choose fine-tuned BERT model as the victim of both the classification and the inference tasks. Since BERT has achieved good results on a variety of NLU tasks and has been proven to be one of the most representative pre-trained transformers (Devlin et al., 2019).

Span generation model To generate variable-length contextualized spans, we choose T5 (Raffel et al., 2020) for the generation. T5 is a representative encoder-decoder language model that can predict the missing words within a corrupted piece of text, benefiting from the fill-in-the-blank pre-training. Also, the large pre-training dataset C4 renders T5 rich prior knowledge to enable the diversity and the high-quality of the generated spans.

Constraints To achieve human imperceptibility and semantic preservation of the adversarial text, we impose constraints on the word perturbation rate and semantic similarity, as defined in Section 3.3. Following previous practices (Jin et al., 2020; Li et al., 2021), we set the thresholds of word perturbation rate and semantic similarity respectively for each dataset, with details shown in Appendix A.

Settings and Computation Cost All results are derived from a single run since there is no randomness in our model. The maximum size of the sliding window is set as 3. In our implementation, we apply SpaCy (Honnibal and Montani, 2017), NLTK (Loper and Bird, 2002) for text manipulation. We run ValCAT on Intel Xeon E5-2690 2.6GHz Processor with V100 GPU. Averagely it takes 34 secs to generate a successful adversarial sample.

3.2 Dataset and Baselines

To investigate the effectiveness of ValCAT on different types of text, we evaluate it on multiple English datasets. We randomly sample 1000 instances from each of the following datasets: three for text classification, i.e., *AG News*, *Yelp Polarity* and *IMDB*; and three for natural language inference, i.e., *SNLI*, *MNLI* and *QNLI*, with detailed information shown in Appendix B. To prove the effectiveness of ValCAT comprehensively, we compared ValCAT with several state-of-the-art word-level black-box attacks, i.e., *TextBugger*, *TextFooler* and *BERT-Attack*. Details of these attacks are shown in Appendix C. Note that, all the datasets and baseline models are publicly available and are used in accordance with their usage specifications.

Dataset	Algorithm	Orig Acc	Atk Acc↓	Suc↑	PPL↓	Sim↑	Pert↓	GErr↓
AG News (PPL=98.4)	ValCAT	94.4	**35.3**	**62.6**	**136.2**	**0.922**	16.1	**0.39**
	BERT-Attack		47.5	49.7	326.9	0.873	17.3	1.12
	TextFooler		44.8	52.5	418.6	0.883	**15.7**	1.37
	TextBugger		62.0	34.3	500.7	0.886	19.7	2.78
Yelp Polarity (PPL=71.1)	ValCAT	98.3	**6.8**	**93.1**	**81.6**	**0.950**	11.6	**0.11**
	BERT-Attack		20.2	79.5	162.9	0.881	14.1	0.15
	TextFooler		27.7	71.8	174.5	0.890	**11.2**	0.33
	TextBugger		53.7	45.4	255.4	0.876	16.1	2.17
IMDB (PPL=58.8)	ValCAT	94.6	**12.8**	**86.5**	**65.6**	**0.977**	6.7	0.09
	BERT-Attack		18.3	80.7	98.0	0.950	7.8	**0.08**
	TextFooler		30.2	68.1	95.6	0.956	**5.9**	0.24
	TextBugger		55.9	40.9	123.2	0.952	8.7	1.49
SNLI (PPL=68.0)	ValCAT	89.8	**10.6/9.7**	**88.2/89.2**	**90.9/78.6**	**0.854/0.840**	**21.3**/22.9	**0.22**/0.15
	BERT-Attack		34.8/20.5	61.2/77.2	184.5/116.9	0.734/0.741	24.8/**19.3**	0.31/**0.09**
	TextFooler		41.7/23.3	53.6/74.0	235.9/146.8	0.734/0.745	24.4/19.4	0.64/0.22
	TextBugger		55.5/34.9	38.2/61.1	374.0/190.9	0.728/0.754	31.0/25.1	1.80/0.69
MNLI (PPL=79.5)	ValCAT	82.7	**7.3/2.5**	**91.2/97.0**	**93.5/89.5**	**0.879/0.869**	**18.5**/20.4	0.19/0.15
	BERT-Attack		22.4/17.7	72.9/78.6	172.9/146.9	0.754/0.764	22.6/18.9	**0.09/0.12**
	TextFooler		28.9/21.4	65.1/74.1	228.1/183.4	0.754/0.770	22.1/**18.2**	0.63/0.38
	TextBugger		41.1/34.7	50.3/58.0	317.6/218.2	0.744/0.771	27.3/22.4	1.98/1.05
QNLI (PPL=66.1)	ValCAT	90.0	**18.7/6.4**	**79.2/92.9**	**70.4/87.5**	**0.828/0.892**	27.8/**18.4**	**0.13/0.18**
	BERT-Attack		32.5/24.7	63.8/72.6	101.2/215.5	0.734/0.744	24.2/26.5	0.31/0.63
	TextFooler		41.0/34.3	54.4/61.9	120.3/252.0	0.757/0.754	**20.7**/24.0	0.51/1.11
	TextBugger		50.4/44.1	43.9/51.0	152.6/388.8	0.753/0.744	27.4/31.8	1.25/3.02

Table 2: Performance of ValCAT in attack success rate (Suc), perplexity (PPL), semantic similarity (Sim), word perturbation rate (Pert) and grammar error (GErr). Bold font indicates the best performance for each metric. ↑ (↓) represents that the higher (lower) the better. The original PPL for each dataset is indicated in parentheses under its name.

3.3 Automatic Evaluation

Metrics We evaluate the effectiveness of ValCAT based on the following metrics:

• *Attack success rate (Suc)*: is the percentage of attack samples successfully interfered with the victim model's prediction over the text dataset.

• *Perplexity (PPL)*: is an automatic metric to evaluate the probability of a sentence appearing in a natural corpus. So PPL can reflect the text's natural fluency, the lower the better. We use GPT-2 (Radford et al., 2019) for this calculation.

• *Semantic similarity (Sim)*: is the cosine similarity between the original text and adversarial text represented by Universal Sentence Encoder (USE) embedding (Cer et al., 2018).

• *Word perturbation rate (Pert)*: is the proportion of the modified words over the original text. To evaluate the perturbation rate on variable-length perturbations more accurately, we design separate calculations for these two perturbation approaches. For *Replace* operation, the number of modifications is $max(l_t, l_a) - l_{LCS}$, where l_t, l_a, and l_{LCS} are the lengths of the target span, the adversarial span, and their longest-common-subsequence (LCS), respectively. For *Insert* operation, the number of modifications is the length of the inserted span.

• *Grammar error (GErr)*: is the incremental grammatical error of the adversarial text relative to the original. We use LanguageTool [1] to count the grammatical errors within a text.

Results The main experimental results are revealed in Table 2. ValCAT outperforms baselines on multiple datasets in both classification tasks and inference tasks. Compared with BERT-Attack, the best baseline method, ValCAT achieves higher success rates in all experiments and the improvement ranges from 5.9% even to 18.8%. This is attributed to ValCAT's ability to apply variable-length contextualized transformations on spans at any position, which largely extends the form of perturbations. In addition, ValCAT achieves the highest semantic similarity with the original text in all tasks. The per-

[1] https://www.languagetool.org/

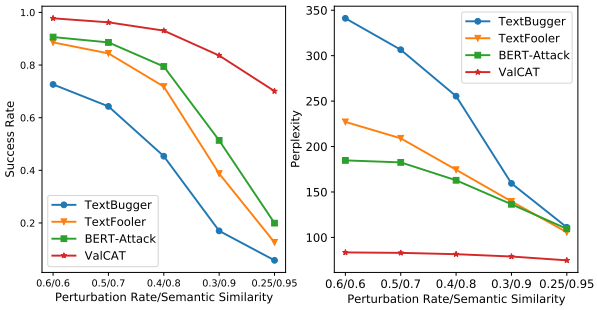

Figure 2: Success rate and perplexity of each attack method under the constraints of different perturbation rates and semantic similarities.

plexity of the adversarial text generated by ValCAT is far superior to the baselines. It is basically only 50% of that of BERT-Attack and is almost consistent with the original text. This indicates that the adversarial text generated by ValCAT is with high fluency. The underlying reason is that multi-word transformations render the language model enough space to generate the adversarial spans with approximated distribution with natural language. At the same time, ValCAT reaches the lowest average increase in the number of grammatical errors on five out of nine attacks, with only a slight disadvantage on the natural language inference tasks.

In general, ValCAT generates adversarial samples that achieve more promising attacks and are more imperceptible to humans. Figure 2 shows the performance of ValCAT and baselines on the Yelp dataset under different constraint settings. As we can see, the success rate of ValCAT decays most slowly as the limits of perturbation rate and semantic similarity increase, still achieving a success rate of almost 70% under the strongest limits. Moreover, ValCAT can maintain a low level under any constraint, however, other attacks have to sacrifice attack success rate to maintain their fluency.

3.4 Human Evaluation

Design We evaluate the quality of the adversarial samples from three perspectives: fluency&grammaticality, semantic similarity, and label consistency. The first two metrics are evaluated by ratings, while the third one lets the annotator categorize the texts into a set of labels. Five annotators with bachelor's degrees performed the evaluation on two datasets, AG News and MNLI. All annotators were informed and consented to the use of the annotation, and were paid with remuneration higher than the regional average. To construct the dataset for human evaluation, we randomly select 100 samples each from the two datasets, whose adversarial results generated by both ValCAT and BERT-Attack can mislead the victim model to make the same wrong classification decision among multiple classes. For the evaluation of fluency&grammaticality, each original text and its two adversarial samples are presented in one group in a shuffled order. The annotators are asked to rate them in terms of fluency and grammaticality. For the evaluation of similarity, given the original text, the judges need to separately evaluate its semantic similarity with the two adversarial samples presented in random order. The above two tasks ask the annotator to rate the text or the text pairs from 1-5. In each questionnaire, we give explicit criteria and clear examples for every score. For the evaluation of label consistency, we mix all the original texts with the adversarial samples and let the annotators label the category of them, e.g., business or technology.

Dataset	Metric	ValCAT	Original	BERT-Attack
AG News	F&G	3.37	3.97	2.98
	Sim	3.92	-	3.41
	Acc	63.0	65.0	62.0
MNLI	F&G	3.97	4.30	3.49
	Sim	3.28	-	2.83
	Acc	46.0	65.0	41.0

Table 3: Results of human evaluation in fluency&grammaticality, semantic similarity and label consistency (accuracy).

Results For the two rating tasks, we normalize the rates of the annotators and calculate the average score, while for the labeling task we take the majority vote as the final result. As shown in Table 3, ValCAT is rated with much higher fluency&grammaticality scores than BERT-Attack on both datasets. Also, it achieves obviously higher semantic similarity. In terms of labeling accuracy, ValCAT slightly outperforms BERT-Attack, still indicating that ValCAT makes smaller changes to the meanings of the text from the aspect of human perception. All these results demonstrate the superior quality of the adversarial samples generated by ValCAT.

4 Analysis

4.1 Ablation Study

We evaluate different attack strategies of ValCAT as in Table 4. *3-Many* results in the lowest perplexity, for it renders the language model more space to gen-

erate a proper adversarial span. It also requires the smallest number of queries, and we speculate the reason as it perturbs more words in a single transformation. For the same reason, *3-Many* is less likely to succeed since it's easier to reach the perturbation constraint. As expected, the combination of multiple types of *Replace* operation facilitates the attack success rate while increasing the number of queries. The *Insert* operation achieves high similarity while requiring a large number of queries. A comprehensive utility of all operations (i.e., ValCAT) achieves the highest success rate with the lowest perturbation rate, which fully demonstrates the advantages of the transformation diversity.

Algorithm	Suc↑	PPL↓	Sim↑	Pert↓	Query↓
ValCAT	**93.1**	81.6	0.950	**11.6**	673
1, 2, 3-Many	90.8	81.1	0.938	13.4	490
1, 2-Many	91.7	82.6	0.939	12.6	431
1-Many	87.9	84.4	0.933	12.0	340
2-Many	86.1	81.5	0.941	13.3	325
3-Many	80.7	**77.1**	0.938	16.0	**266**
Insert	86.4	86.9	**0.953**	14.4	799
BERT-Attack	79.5	162.8	0.881	14.1	449

Table 4: Results of ablation study. *n-Many* is a type of *Replace* operation, which means that a span of length n is replaced by an adversarial span.

4.2 Impact of Candidate Numbers

From the results under different numbers of candidates constraints shown in Figure 3, we observe that the attack success rate is higher when there are more candidates. Since we greedily select the adversarial span with the greatest impact on the decision, as the number of candidates increases, spans with low occurrence probability can be added to the text, making the perplexity higher. This is especially true for difficult-to-attack datasets like AG News.

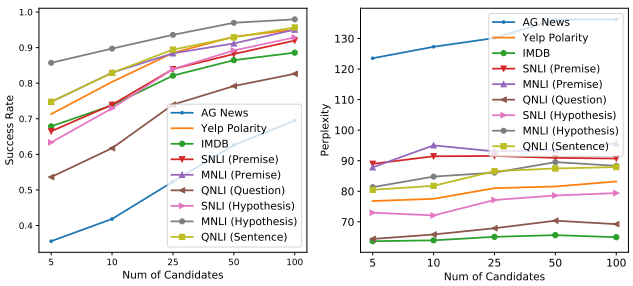

Figure 3: Success rate and perplexity of each dataset under different candidate numbers.

4.3 Properties of Transformations

We observe the properties of the transformations in the successful adversarial samples from three perspectives: type of transformation, length of span, and POS (Part-of-Speech) of span. Averagely, 62% of the transformations in each sample are *Replace* operations and 38% are *Insert* operations, indicating that *Replace* operations are more effective in most cases. As shown in Table 10, the replaced spans are most likely the longer ones, while the adversarial spans generated by both two operations are rather short. Additionally, Table 11 shows that the adversarial spans of *Replace* operation share identical POS distribution with the replaced span, which guarantees the fluency of the adversarial text. The *Insert* operation tends to use adverbs and adjectives, which is consistent with the human intuition of inserting words into text.

4.4 Adversarial Training

We investigate how adversarial training can mitigate our attack, as well as the power of our adversarial samples for adversarial training as a general defense to adversarial attacks. We randomly sample 10,000 instances from the *Yelp* training dataset and apply ValCAT to generate adversarial samples. These adversarial samples and their intermediate results, approximately 60,000 texts, are then used to fine-tune the BERT under the gold labels.

After adversarial training, the test accuracy of the victim model decreases from 98.3% to 98.0%. As shown in Table 5, the attack success rate of ValCAT drops by about 20%, while both the word perturbation rate and the number of queries have significantly increased, indicating that the victim model is hard to attack after the adversarial training. Compared to the results of BERT-Attack, the adversarial samples generated by ValCAT have an excellent generalization effect on adversarial training, with a higher success rate.

Algorithm	Victim	Att Acc↓	Suc↑	Pert↓	Query↓
ValCAT	Original	6.8	93.1	11.6	673.3
	Adv Train	27.0	72.4	14.0	884.2
BERT-Attack	Original	20.2	79.5	14.1	449.8
	Adv Train	37.7	61.5	13.8	430.6

Table 5: Results of Adversarial Training from ValCAT and BERT-Attack.

4.5 Generalization

We evaluate the generalization of ValCAT in two aspects: 1) attack on other victim models and 2)

transferability.

Victim	Orig Acc	Att Acc↓	Suc↑	PPL↓	Sim↑	Pert↓
BERT	98.3	6.8	93.1	81.6	0.950	11.6
RoBERTa	99.1	13.3	86.6	81.2	0.952	11.7
LSTM	95.3	0.7	99.3	78.5	0.961	9.2
wordCNN	95.4	1.1	98.8	78.9	0.957	10.0

Table 6: Results of ValCAT on other victim models.

Attack on other victim models We try to attack other common language models on the Yelp dataset using ValCAT. Table 6 shows that LSTM and wordCNN, two traditional NLP models, are extremely vulnerable to ValCAT with a success rate of around 99%. Interestingly, RoBERTa shows better robustness, with a 6.5% decrease in attack success rate relative to BERT, which we speculate is due to its more optimized pre-training approach.

Transferability We evaluate transferability of generated adversarial texts on all datasets. Specifically, we utilize attack samples generated by different target models which successfully interfered with the victim model, to attack other tested models. Results in Table 7 shows that, attack samples generated by transformer models (BERT and RoBERTa) could cause similar accuracy decay on traditional NLP models (LSTM and wordCNN), and vice versa. It indicates that transformer models (BERT and RoBERTa) have better transferability than traditional NLP models (LSTM and wordCNN).

	BERT	RoBERTa	LSTM	wordCNN
BERT	-	73.4	73.9	76.1
RoBERTa	68.9	-	72.8	74.2
LSTM	84.3	88.8	-	71.8
wordCNN	83.4	87.3	60.5	-

Table 7: Attacked accuracy of Transferability. Rows are target models used in generating adversarial samples, and columns are tested models applied generated adversarial samples.

4.6 Limitation

ValCAT makes the best effort to improve the quality of the generated adversarial samples, including variable-length span ranking and multiple adversarial text generation strategies. However, all these efforts result in a larger number of queries to the victim model, compared to word-level adversarial substitutions. According to the results of the automatic and human evaluation, such an increase in the computation expense is tolerable, considering the significant improvement in adversarial sample quality and attack success rate. Furthermore, a simplified version of ValCAT, *1-Many* (See Table 4), outperforms the best baseline model in all aspects, including the number of queries.

5 Related Work

Textual adversarial attacks have been intensively studied (Wang et al., 2019; Zhang et al., 2020). Early on, the character-level attack methods (Ebrahimi et al., 2018; Gao et al., 2018; Li et al., 2019) are widely used, but them would destroy words and are very perceptible (Pruthi et al., 2019). Word-level attacks are now in the spotlight, evolving from substitution based only on the properties of individual words themselves (Zang et al., 2020; Jin et al., 2020; Tan et al., 2020) to perturbing words by multiple strategies with knowledge of the context (Li et al., 2020;Garg and Ramakrishnan, 2020;Li et al., 2021). However, these works only focus on context-sensitive substitution on single word, with absence of many possible forms of perturbations. Limitations on single-word substitution hinders further improvement in attack success rate and fluency. Although, Zhang et al. 2019, Wang et al. 2020 and Huang and Chang 2021 attempt to construct adversarial samples from levels above words by simply merge single-word, they all ignore the interactions between words are in finding important positions.

We propose ValCAT, which takes into account the interactions between words in both ranking important spans and generating variable-length contextualized spans. By introducing *Sliding Window*, ValCAT can find important spans. Based on the mechanism of encoder-decoder, ValCAT can generate variable-length substitution candidates with two perturbation approaches, *Replace* and *Insert*.

6 Conclusion

In this paper, we propose ValCAT, the first variable-length contextualized adversarial attack based on the encoder-decoder language model. ValCAT considers the interaction between words for the vulnerable-position ranking and extends the form and the flexibility of the perturbation. Experimental results on several datasets demonstrate the effectiveness of our model, which outperforms baselines in terms of attack success rate, adversarial example quality, transferability, and robustness against robust training. A comprehensive human evaluation further verifies the high quality of our generated adversarial samples in reality.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. 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)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

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

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Kuan-Hao Huang and Kai-Wei Chang. 2021. Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033, Online. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. 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*, volume 34, pages 8018–8025.

Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. 2016. Adversarial examples in the physical world.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2021. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5053–5069, Online. Association for Computational Linguistics.

J Li, S Ji, T Du, B Li, and T Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. *arXiv preprint arXiv:1907.07355*.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591, Florence, Italy. Association for Computational Linguistics.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin' time! Combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Boxin Wang, Hengzhi Pei, Boyuan Pan, Qian Chen, Shuohang Wang, and Bo Li. 2020. T3: Tree-autoencoder constrained adversarial text generation for targeted attack. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6134–6150, Online. Association for Computational Linguistics.

Wenqi Wang, Run Wang, Lina Wang, Zhibo Wang, and Aoshuang Ye. 2019. Towards a robust deep neural network in texts: A survey. *arXiv preprint arXiv:1902.07285*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Han Xu, Yao Ma, Hao-Chen Liu, Debayan Deb, Hui Liu, Ji-Liang Tang, and Anil K Jain. 2020. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, 17(2):151–178.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080, Online. Association for Computational Linguistics.

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. 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)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

A Constraints

	Perturbation Rate	Semantic Similarity
AG News	0.4	0.8
Yelp Polarity	0.4	0.8
IMDB	0.3	0.9
SNLI	0.6	0.6
MNLI	0.6	0.6
QNLI	0.6	0.6

Table 8: Specific values of constraints of the perturbation rate and semantic similarity used in the main experiments.

B Datasets

We employ three datasets for text classification and three datasets for natural language inference as described below. In our experiments, we randomly sample 1000 records.

• *AG News*: a collection of news articles categorized into 4 types: World, Sports, Business, and Sci/Tech (Zhang et al., 2015).

• *Yelp Polarity*: positive and negative restaurant reviews collected from yelp (Zhang et al., 2015).

• *IMDB*: a dataset of movie reviews for binary sentiment classification (Maas et al., 2011).

• *SNLI*: a collection of human-written English sentence pairs (Bowman et al., 2015). Each sentence pair consists of a premise and a hypothesis, which is necessary to determine whether it is entailment, contradiction, or neutral.

• *MNLI*: a similar dataset to SNLI (Williams et al., 2018), but from a variety of genres.

• *QNLI*: a version of SQuAD which has been converted to a binary classification task (Wang et al., 2018).

C Baselines.

We compare ValCAT with state-of-the-art word-level black-box attacks:

• *TextBugger*: an attack model compounded by five bug generation methods (Li et al., 2019). Such

methods include character insertion, character deletion, character swapping, homograph character replacement, and synonym replacement.

• *TextFooler*: classical adversarial attack algorithm based on synonym substitution (Jin et al., 2020). Candidate synonyms are the closest neighbors of the replaced word in the counter-fitting word embedding space. It limits the cosine similarity and makes the POS consistent.

• *BERT-Attack*: a state-of-the-art contextualized attack model using BERT to fill mask tokens (Li et al., 2020). It serves as a representative of the similar BERT-based algorithms, such as BAE (Garg and Ramakrishnan, 2020) and CLARE (Li et al., 2021).

D Performance of Encoder-Decoder Models

We present the results of generating adversarial spans by several language models in the T5 family in Table 9. $T5_{large}$ and $mT5_{base}$ take a longer time to generate adversarial span due to their larger capacity. The higher perplexity of $T5v1.1_{base}$ is, we conjecture, resulted from its specifically different structure. In short, $T5_{base}$ is good enough for adversarial span generation.

Algorithm	**Suc↑**	**PPL↓**	**Sim↑**	**Pert↓**	**GErr↓**	**Time↓**
$T5_{base}$	93.1	81.6	0.950	**11.6**	0.11	**0.052**
$T5_{large}$	90.3	84.0	0.948	12.7	0.12	0.069
$T5v1.1_{base}$	92.3	101.8	0.946	12.4	0.36	0.055
$mT5_{base}$	**94.2**	84.6	**0.949**	12.3	**0.06**	0.070

Table 9: Encoder-Decoder Language Model

E Properties of Transformations

We observe the properties of the transformations in the successful adversarial samples from three perspectives: type of transformation, length of span, and POS (Part-of-Speech) of span. Results shown as follows:

	1	2	3	4	5
Replaced Span	26.2%	35.8%	38.0%	-	-
Span for *Replace*	33.1%	35.0%	24.6%	6.4%	0.9%
Span for *Insert*	37.5%	35.3%	27.2%	-	-

Table 10: Transformation Length Proportion

Replaced Span	Span for Replace	Span for Insert
ADJ: 7.6%	ADJ: 8.4%	ADV: 12.4%
NOUN: 6.3%	VERB: 6.7%	ADJ: 6.7%
VERB: 5.0%	NOUN: 5.8%	NOUN: 2.2%
ADV: 3.8%	ADV: 5.5%	ADJ-NOUN-PUNCT: 1.9%
VERB-ADV: 2.3%	ADJ-PUNCT: 1.8%	VERB: 1.9%
AUX-ADV: 1.9%	ADV-ADJ: 1.7%	ADV-ADV: 1.6%
ADJ-PUNCT: 1.5%	AUX-ADV: 1.7%	ADJ-PUNCT: 1.5%
ADV-VERB: 1.5%	VERB-ADV: 1.5%	VERB-ADV: 1.4%
PRON-VERB: 1.4%	NOUN-PUNCT: 1.4%	PRON-VERB: 1.4%
ADV-ADJ: 1.3%	ADJ-NOUN: 1.4%	ADV-PUNCT: 1.2%

Table 11: Transformation POS Proportion