XVD: Cross-Vocabulary Differentiable Training for Generative Adversarial Attacks

Anonymous submission

Abstract

An adversarial attack to a text classifier consists of an input that induces the classifier into an incorrect class prediction, while retaining all the linguistic properties of correctly-classified examples. A popular class of adversarial attacks exploits the gradients of the victim classifier to train a dedicated generative model to produce effective adversarial examples. However, this training signal alone is not sufficient to ensure other desirable properties, such as similarity to non-adversarial examples, linguistic fluency, grammaticality, and so forth. For this reason, in this paper we propose a novel training objective which leverages a set of pretrained language models to promote such properties in the adversarial generation. A core component of our approach is a set of vocabulary-mapping matrices which allow connecting the generative model to any victim or component model of choice, while retaining differentiability end-to-end. The proposed approach has been tested in an ample set of experiments including six text classification datasets, two victim models, and four baselines. The results show that it has been able to produce effective adversarial attacks, outperforming the compared generative approaches and proving highly competitive against established token-replacement approaches.

1 Introduction

Text adversarial attacks are subtly manipulated inputs to a machine learning model that have the intent of causing erroneous predictions. These manipulations can drastically alter a model's behaviour and represent a significant challenge for the entire field of machine learning. In the context of text classification, adversaries employ a wide range of techniques, from simple token alterations to full training of generative models, each aiming to exploit the model's weaknesses while also preserving the textual content's semantic coherence and grammaticality.

The most prevalent adversarial attack strategy is the *token-based* approach, where adversarial examples are crafted through a sequence of token modifications — replacements, additions, or deletions — guided by search methods like beam search, all while maintaining a series of constraints (Morris et al., 2020). These attacks are simple and effective, but the search method must be run for each example, and the process can prove very time consuming (Yoo et al., 2020). Conversely, *generative* approaches train a text-to-text model to directly produce transformations from original to adversarial examples. These attacks, though less studied, can explore a more expansive range of transformations than token-based attacks, and at inference time can rapidly generate a diverse and intriguing array of adversarial examples. The approach is flexible, with a range of text-to-text models able to be used for this purpose; examples include Generative Adversarial Networks (GANs) (Zhao et al., 2018), paraphrasers (Iyyer et al., 2018), autoencoders (Xu et al., 2021), or style transfer models (Qi et al., 2021). The main drawback of the generative approach is that the model must be trained to generate effective attacks, which can be challenging due to the difficulty of manual supervision and the lack of straightforward training approaches (Wong, 2017).

Adversarial attacks also differ in the amount of assumed access to the classification model (often called the *victim model*). One common assumption is the *black-box* scenario, where attacks only require access to the victim model's outputs, or sometimes the logits (Biggio and Roli, 2018). The opposite is the *white-box* scenario, where the adversary assumes full information access, including gradients, data, loss functions, and model parameters — effectively, the worst-case scenario for an attacked system (Biggio and Roli, 2018). These assumptions may seem hard to meet in practice, but increasingly they reflect realistic scenarios due to the widespread adoption of publicly available machine learning models (such as those found on the Hugging Face Model Hub[1]). On the other hand, developers can use white-box attacks to identify and fix vulnerabilities in their model. In short, studying white-box attacks remains critical.

An intuitive approach for training a white-box generative attack is to link the generative model to the victim model, so as to use the feedback from the victim model as a training signal. However, this signal is not sufficient to ensure all the other properties required of a satisfactory adversarial attack, such as fluency, grammaticality, closeness to non-adversarial examples, and so forth. For this reason, in this paper we propose leveraging a suitable suite of pretrained language models to encourage such properties at training time. During the forward pass, our generative model receives an original example in input and generates a "soft token" prediction of adversarial example in output, which is then passed to the victim and downstream models for their processing. Softening the prediction ensures that the entire pipeline remains end-to-end differentiable, and able to leverage the training objectives of the downstream modules as an effective adversarial attack loss function.[2] The parameters of the generative model are then updated in the backward pass, while the parameters of the other models are all kept frozen. After training, the generative model is able to generate not only one, but multiple adversarial candidates per original example, simply by using conventional beam search or any other decoding method.

An immediate challenge to this approach is that the use of soft predictions to permit overall differentiability requires the alignment of the models' vocabularies, which is not easy to ensure. The simplest workaround is to constrain all the models to share the same vocabulary and tokenisation algorithm. However, this severely limits the choice of pretrained models. Another possible approach is to restrict the vocabularies of all models to their tokens in common (Song et al., 2021). However, this may majorly limit the expressiveness and articulation of the learned adversarial strategies. Overall, the vocabulary alignment between language models still seems to be a partially unresolved issue in the literature.

[1]https://huggingface.co/models

[2]The generative model cannot directly pass text to the other models while keeping the training signal differentiable, as it needs either sampling from the token distribution or taking an $argmax$ — both of which are non-differentiable operators.

For this reason, in this paper, we propose an original approach for training a cross-vocabulary, differentiable white-box generative attack that is able to circumvent this restriction. The core components of the proposed approach — nicknamed XVD, from 'cross-vocabulary differentiable' — include: 1) the use of a suitable set of pretrained language models to provide training signals to the adversarial attack generator; 2) the adoption of soft predictions to ensure end-to-end differentiability, and 3) a set of sparse vocabulary-mapping matrices that map tokens between the vocabulary of the generative model and those of the victim and downstream models, allowing complete freedom in the choice of models. The generative model is then trained using a highly configurable, overall loss function that balances text quality with attack strength. In the experiments, the proposed approach has been compared against four baseline methods on six text classification datasets and two victim models. The results show the effectiveness of the proposed approach at consistently generating high-quality adversarial examples across the range of datasets and victim models. In addition, a comprehensive ablation analysis highlights the contributions of the various components and suggests ways for future improvements.

In summary, our paper makes the following contributions:

1. a novel approach for training generative white-box attacks, based on training signals from a set of pretrained language models and a fully differentiable loss function;
2. a vocabulary-mapping module which grants interoperability to any chosen combination of generative, victim or loss component models;
3. extensive experiments over six text classification datasets and two victim models that give evidence to the effectiveness of the proposed approach;
4. a comprehensive ablation and sensitivity analysis that delves into its benefits and limitations.

2 Related Work

White-box token-based attacks date back to at least the work of Papernot et al. (2016). Typically, these attacks leverage the gradient signal of the victim model in two main ways. The first is to rank token importance in the original sentence, thus identifying promising attack targets, as demonstrated in

Wallace et al. (2019). The second is to aid in selecting token transformations that best meet adversarial criteria, as shown in various character-level and word-level attacks (Ebrahimi et al., 2018; Zhang et al., 2019; Liang et al., 2018).

The differentiable model-cascading approach described in Section 1 has been explored by several other studies. For instance, Xu et al. (2021) have used an autoencoder as the generative model and examined several modifications to its training process, such as label smoothing and copy mechanisms, to enhance the quality of the generated examples. Wang et al. (2020) have proposed incorporating a downstream model which allows the generative model to control the topic of its generated adversarial candidates at inference time. In contrast, Song et al. (2021)'s approach is based on training the generator to generate trigger phrases that, when concatenated to an input sentence, induce misclassification in the victim model. In turn, Guo et al. (2021) have proposed learning an example-dependent matrix of token probabilities, which at inference time is sampled to generate adversarial examples. However, none of these approaches has proposed a systematic and configurable solution for training the generative model to satisfy all the desirable properties of an adversarial attack.

In terms of the vocabulary-alignment issue, the works of Xu et al. (2021), Wang et al. (2020) and Guo et al. (2021) have all acknowledged the problem, but only implemented the shared-vocabulary scenario. Conversely, Song et al. (2021) have constrained the generative model to only output the common tokens of all vocabularies. As observed in the Introduction, neither of these solutions can be regarded as satisfactory.

3 Proposed Approach

3.1 Overview

We aim to fine-tune a generative model g, with parameters θ and vocabulary V_g, to generate adversarial examples for victim model v, with vocabulary V_v. The approach includes two additional component models for the training objective: a semantic similarity model, s, of vocabulary V_s, and a natural language inference (NLI) model, n, of vocabulary V_n. The parameters of models v, s and n are all fixed, while those of g are the target of the proposed training approach. The complete setup is shown in Figure 1.

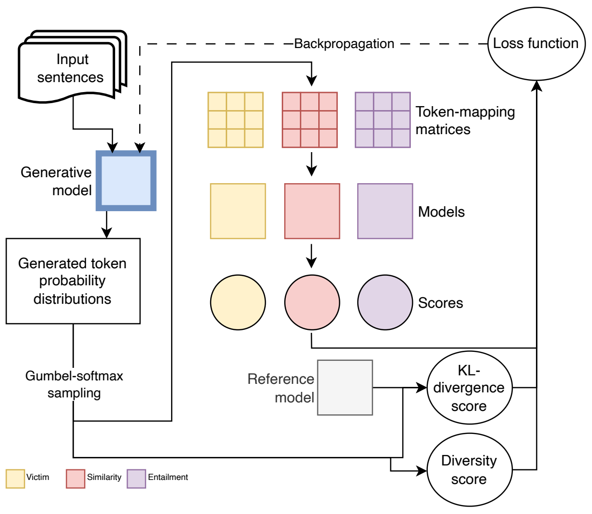

Figure 1: The training approach. The loss function is composed of scores from a number of cascaded models (depicted by squares), a KL divergence score using a reference model, and a diversity score. The parameters of the generative model are updated using standard backpropagation.

3.2 Training

We initialise the generative model, g, with a pre-trained paraphrase model as it is already capable of a range of diverse, semantic-preserving transforms. Given an original example x, we employ g to generate an example x' of length T and its corresponding sequence of token probability distributions, which forms a matrix P with dimensions $T \times |V_g|$.

We then use the token probability distribution matrix, a vocabulary-mapping matrix, and the token embedding matrix of the downstream model to create a weighted average of token embeddings, allowing us to retain the desirable differentiability. Formally, for any component model V_i, with $i \in \{v, s, n\}$, the respective weighted embeddings W_i are computed as:

$$W_i = PM_iE_i$$

where E_i is the token embedding matrix of model i, and M_i is a vocabulary-mapping matrix (described in Section 3.3) that maps V_g, the vocabulary of P, to V_i, the vocabulary of model i.

Additionally, to control the diversity of the generated embeddings, we use the Gumbel-softmax reparametrisation trick (Jang et al., 2017), replacing P with a sampled matrix P_b that incorporates Gumbel(τ) noise, where τ is a chosen temperature parameter. Values of $\tau > 1$ make the samples more evenly distributed, while values < 1 concentrate them towards a one-hot distribution. Prior research has also used this technique to increase exploration during training (Xu et al., 2021; Wang et al., 2020).

After computation, the weighted embeddings $W_i, i \in \{v, s, n\}$ are fed into the component models, and their output scores are used in the loss function (Section 3.4). The generative model's parameters are updated via standard backpropagation.

3.3 Vocabulary-mapping matrices

We construct a vocabulary-mapping matrix, M_i, to map tokens from the generative model's vocabulary, V_g, to the vocabulary of each component model, noted as V_i hereafter. The matrix has shape $|V_g| \times |V_i|$, and each row is a probability distribution that represents the one-to-many token mapping, with values summing to 1. This is a large matrix, and to save space we have implemented it as a sparse matrix.

Mapping tokens between vocabularies, each possibly built with a different tokenisation algorithm, is not straightforward. In our implementation, the generative model's tokeniser uses SentencePiece (Kudo and Richardson, 2018), while the tokenisers of all the component models use WordPiece (Wu et al., 2016). We match tokens where possible using string matching rules, and use the component model tokeniser to create mappings for the remainder. The process is described fully in Appendix B.

3.4 Loss function

In accordance with the definition provided by Michel et al. (2019), our aim is to create adversarial examples that successfully tweak the predicted labels, yet ensure retention of the original text's meaning alongside linguistic acceptability. To this end, our training objective, $l(x, x')$:

$$\begin{aligned} l(x,x') = \alpha_v t(v(x,x'),\beta_v) + \\ \alpha_s t(s(x,x'),\beta_s) + \\ \alpha_e t(e(x,x'),\beta_e) - \\ \alpha_{KL} t^*(D_{KL}(x,x'),\beta_{KL}) \end{aligned} \tag{1}$$

integrates multiple components as follows:

- $v(x, x')$ represents the 'victim model score', a measure of how much the classifier's confidence in the correct class drops when replacing x to x' in input.

- $s(x, x')$ is the 'similarity score' between x and x', which is based on the cosine similarity of their sentence embeddings as computed by a pretrained Sentence-BERT model (Reimers and Gurevych, 2019)

- $e(x, x')$ is the 'entailment score', which measures the probability that x's ground-truth label is retained by x', and is approximated with the probability of x entailing x' using a pretrained NLI model.

- D_{KL} is the Kullback-Leibler (KL) divergence between the token probabilities output by the fine-tuned generative model, noted as g, and those of a reference model, noted as g^*, and taken as the initial pretrained model. D_{KL} is defined as:

$$\begin{aligned} D_{KL} = \frac{1}{T}\mathbb{E}_{x\sim\mathcal{D},x'\sim g(x;\theta)}[\log p_g(x'|x) - \\ \log p_{g^*}(x'|x)] \end{aligned} \tag{2}$$

  where the generated sequence length, T, is used to normalise the divergence to prevent longer sequences being unfairly penalised. This term encourages the fine-tuned distribution to not deviate excessively from the initial, preserving the generative properties of g^*.

- t and t^* are threshold clipping operators, with $t(a, \beta) = a$ if $a < \beta$, and 0 otherwise, and $t^*(a, \beta) = a$ if $a > \beta$, and 0 otherwise. As such, α and β are hyperparameters that control each term's contribution.

The training objective $l(x, x')$ is incorporated into the final *batch-level* loss, L, defined as:

$$L = -\left(\frac{1}{|B|}\sum_{(x,x')\in B} l(x,x')\right) + \alpha_d\, d(B) \tag{3}$$

where $d(B)$ is a batch-level diversity score, and α_d its corresponding coefficient. To compute $d(B)$, we first compute the mean of the token embeddings for each generated sentence within batch B. We then calculate the cosine similarity between each pair of mean embeddings using the same model as the similarity score, and compute $d(B)$ as the average, with lower values indicating more diversity. We found that the inclusion of this term can effectively prevent mode collapse and encourage variety in the generated examples.

Each term in the loss function is differentiable, allowing for efficient minimisation via backpropagation. The coefficients can be adjusted to prioritise different objectives, such as attack strength or fluency.

3.5 Validation and early stopping

During fine-tuning, it is important to enforce early stopping to prevent text quality degradation due to over-training. To this end, during validation we generate eight candidates per original, using diverse beam search (Vijayakumar et al., 2016). For each candidate, we check if its scores from Equation 1 surpass the corresponding β thresholds (or, in the case of the KL divergence, fall below). The validation metric we adopt is the proportion of attacks that have at least one candidate that successfully passes all checks. We calculate the validation metric multiple times per epoch, and halt the training process once it fails to improve over a patience interval, as standard for early stopping.

4 Experimental setup

4.1 Datasets

We have conducted experiments over six diverse English text classification datasets (Table 1). The Hate Speech dataset (HS) classifies offensive language in tweets as hate speech, offensive language, or neither (Davidson et al., 2017); the Text REtrieval Conference (TREC) question-type classification dataset (Li and Roth, 2002) and the SUBJ dataset (Pang and Lee, 2004) discriminate between objective and subjective sentences; the Rotten Tomatoes (RT) (Pang and Lee, 2005) and Financial PhraseBank (FP) (Malo et al., 2014) datasets are sentiment analysis datasets of movie reviews and financial news, respectively; and the Emotion dataset classifies text fragments as one of six basic emotions (Saravia et al., 2018). These datasets have been chosen for their variety and attackable short snippets, with concise statistics and examples presented in Table 1. Further details are provided in Appendix A.

4.2 Models

We have used T5-Base (Raffel et al., 2020), which uses SentencePiece for tokenisation, as our generative model, g. We have evaluated attacks on two victim models: a DistilBERT model (Sanh et al., 2019) and an ELECTRA-trained model (Clark et al., 2020), both of which use WordPiece for tokenisation. Each victim model has been fine-tuned on the given dataset prior to being subjected to attacks. Full details of all models (including the semantic similarity and entailment models) are provided in Appendix A.

4.3 Baselines

To comparatively evaluate the performance of our model we have used four established baseline attacks, all included in the comprehensive OpenAttack adversarial attack library of Zeng et al. (2021). TextFooler (Jin et al., 2019) and BERTAttack (Li et al., 2020) form the first set of baselines; both are token-replacement attacks that replace individual tokens sequentially in a constrained optimisation process. We have also included two generative attacks that, like our approach, generate adversarial candidates at inference time. The first is a GAN approach (Zhao et al., 2018), and the second is an adversarial paraphraser, named SCPN (Iyyer et al., 2018), that generates syntactically controlled paraphrases.

4.4 Candidate selection

At inference time, our fine-tuned model is capable of generating, in principle, an unlimited number of candidates per input example. Nevertheless, for the purpose of fair comparison with the baselines outlined in 4.3 that return a single adversarial example per input, we have opted to select only one candidate also from our model.

We begin with the use of diverse beam search (Vijayakumar et al., 2018) to create n candidates for each original example. (A sensitivity analysis of n is presented in Section 6.1.) We then compute a 'quality score' for each candidate as $s(x, x') + e(x, x') - D_{KL}(x, x')$, which represents a rough balance of our text-quality objectives. From these scored candidates, we select those that have managed to flip the ground-truth label. Within this subset, we select the candidate with the highest score amongst those that satisfy all validation checks (Section 3.5). If none meets these requirements, the highest-scoring candidate is chosen instead.

4.5 Evaluation metrics

As an obvious preamble, no ground-truth reference exists for adversarial candidates, and therefore the evaluation has to be orchestrated with adequate unsupervised metrics. To this aim, we have used five evaluation metrics over the test set of each dataset,

Dataset	N (trn/val/tst)	Classification task	#cls	Examples
HS	8k/1k/1k	hate speech detection	3	"I can be very vengeful. Don't be a [..]" (offensive language)
TREC	4k/1k/0.5k	type of question	6	"When did beethoven die?"(num)
SUBJ	2k/0.5k/0.5k	(subject/object)ivity	2	"...routine, harmless diversion and little else." (subj)
RT	3.5k/0.5k/0.5k	sentiment (movies)	2	"A moving and not infrequently breathtaking film." (pos)
FP	1.5k/0.2k/0.2k	sentiment (financial)	3	"Operating profit was EUR 11.4 mn, up from [...]" (pos)
Emotion	10k/1k/1k	emotion detection	6	"i be made to feel rotten"(sad)

Table 1: Statistics and examples from the datasets used. Column N shows the approximate number of examples in each train/validation/test split, and #cls is the number of classes of the dataset.

and reported the median values in Table 2. The first metric, referred to as *Flip*, is the proportion of instances where the ground-truth label of the original example, predicted correctly by the victim model, has flipped in the prediction for the candidate. The next three—*Sim, Flu*, and *Ent*— are text quality metrics, and are only computed for examples that have flipped. To assess the semantic similarity between the original and the candidate (*Sim*), we have used BERTScore F1 (Zhang et al., 2020); to assess the fluency of the candidate (*Flu*), we have used BARTScore (Yuan et al., 2021), a fluency proxy that uses the text generation probability of a seq2seq model; and for the entailment (*Ent*), we have used the probability that the candidate does not contradict the original in the entailment model. The last metric — the Validated Success Rate (*VSR*) — is the proportion of examples that have successfully flipped the label and also met minimum thresholds across the three text quality metrics.[3] While all automated metrics have inherent limitations, our choice of metrics is both consistent with prior literature and able to provide a thorough assessment of the quality of the adversarial candidate.

Postprocessing. Before metric calculation, each successful attack has been post-processed to begin with a capital letter, end with a period, and have no whitespace around the last punctuation character.

5 Results

The results from our experiments are reported in Table 2, showing that the proposed approach, XVD, has been able to generate high-quality adversarial examples with notable success rates (VSR). Compared to the generative baseline methods, GAN and SCPN, XVD's performance has proved better for all experimental combinations bar one. XVD has also performed competitively against the best token-replacement baseline, BERTAttack, scoring best for three out of six datasets with the DistilBERT victim model, and for four out of six with the ELECTRA victim model. XVD has also achieved the highest VSR overall (0.87; Emotion dataset). In particular, it has performed the best with both victim models over the TREC dataset, where its flipping rate has proved much higher than that of the other approaches, and over the HS dataset, probably because the token-replacement baselines have struggled to replace its many slang words in the absence of well-defined synonyms. Qualitative examples of the attacks generated by XVD are presented in Table 3.

Overall, the proposed approach has proved very strong at label-flipping (Flip) and at retaining the original label (Ent), while intermediate in the fluency (Flu) and similarity (Sim) metrics. This is mainly due to its much broader generative space compared, in particular, to the token-replacement attacks. The proposed approach is also, by design, able to pursue different trade-offs between these properties, thanks to its highly configurable training objective and generative behaviour. We explore some of these trade-offs in the following section.

6 Ablations

We have measured the performance impact of various parameters within our model through a series of ablation studies, using the Financial PhraseBank dataset as reference and testing each configuration using three random seeds. The results are presented in the following subsections.

6.1 Number of generated evaluation sequences

The number of sequences generated during inference, n, directly controls the attack's search space. As n increases, we expect a rise in the label-flipping rate and, after a point, a decline in text quality metrics. To measure these effects, we have varied n, employing diverse beam search for $n \geq 2$ (with $n/2$ beam groups) and regular beam search for

[3]We have used the (fairly relaxed) thresholds of: ≥ 0.85 *Sim*, ≥ -4 *Flu*, ≥ 0.6 *Ent*.

Victim Model	Attack	Datasets														
		HS					TREC					SUBJ				
		VSR	Flip	Sim	Flu	Ent	VSR	Flip	Sim	Flu	Ent	VSR	Flip	Sim	Flu	Ent
ELECTRA	BertAttack	0.37	0.50	0.96	-1.84	0.95	0.33	0.62	0.95	-2.35	0.67	0.46	0.71	0.95	-1.96	0.90
	TextFooler	0.29	0.53	0.92	-2.94	0.80	0.18	0.44	0.93	-2.74	0.49	0.24	0.55	0.94	-2.91	0.69
	GAN	0.00	0.78	0.79	-6.38	0.33	0.00	0.70	0.84	-6.38	0.09	0.00	0.35	0.81	-6.08	0.24
	SCPN	0.12	0.71	0.84	-4.49	0.49	0.25	0.87	0.90	-3.87	0.48	0.28	0.63	0.89	-3.28	0.68
	XVD (mean)	**0.46**	0.80	0.89	-3.30	0.88	**0.58**	0.99	0.92	-3.22	0.80	**0.63**	0.92	0.89	-3.25	0.90
	XVD (std)	*0.02*	*0.04*	*0.00*	*0.28*	*0.02*	*0.02*	*0.04*	*0.00*	*0.28*	*0.02*	*0.03*	*0.02*	*0.00*	*0.12*	*0.01*
DistilBERT	BertAttack	0.38	0.53	0.96	-1.87	0.94	0.32	0.64	0.94	-2.56	0.69	**0.40**	0.65	0.95	-1.89	0.83
	TextFooler	0.30	0.54	0.93	-2.91	0.79	0.19	0.44	0.93	-2.76	0.52	0.24	0.51	0.94	-2.58	0.64
	GAN	0.00	0.81	0.79	-6.41	0.33	0.00	0.73	0.84	-6.38	0.10	0.00	0.43	0.81	-6.11	0.20
	SCPN	0.13	0.72	0.84	-4.46	0.51	0.27	0.92	0.90	-3.84	0.55	0.23	0.50	0.89	-3.32	0.70
	XVD (mean)	**0.59**	0.82	0.89	-3.14	0.89	**0.37**	1.00	0.89	-3.70	0.65	0.14	0.98	0.82	-4.04	0.79
	XVD (std)	*0.08*	*0.01*	*0.01*	*0.16*	*0.03*	*0.08*	*0.00*	*0.01*	*0.16*	*0.02*	*0.12*	*0.01*	*0.01*	*0.26*	*0.06*

Victim Model	Attack	Datasets														
		RT					FP					Emotion				
		VSR	Flip	Sim	Flu	Ent	VSR	Flip	Sim	Flu	Ent	VSR	Flip	Sim	Flu	Ent
ELECTRA	BertAttack	**0.47**	0.85	0.96	-1.42	0.79	0.32	0.68	0.96	-1.81	0.56	**0.66**	0.90	0.98	-0.99	0.97
	TextFooler	0.34	0.69	0.96	-2.12	0.69	0.33	0.63	0.94	-2.61	0.71	0.53	0.76	0.97	-1.27	0.95
	GAN	0.00	0.39	0.82	-6.00	0.33	0.00	0.39	0.79	-6.24	0.16	0.00	0.68	0.82	-6.43	0.11
	SCPN	0.28	0.66	0.89	-3.44	0.70	0.14	0.40	0.89	-3.52	0.58	0.37	0.72	0.90	-3.26	0.83
	XVD (mean)	0.30	0.81	0.85	-3.32	0.84	**0.72**	1.00	0.89	-3.12	0.94	0.64	0.97	0.90	-3.24	0.88
	XVD (std)	*0.05*	*0.05*	*0.01*	*0.15*	*0.02*	*0.15*	*0.00*	*0.01*	*0.37*	*0.03*	*0.14*	*0.02*	*0.02*	*0.44*	*0.05*
DistilBERT	BertAttack	**0.46**	0.89	0.96	-1.40	0.71	**0.43**	0.79	0.96	-1.77	0.82	0.67	0.88	0.98	-0.97	0.98
	TextFooler	0.36	0.70	0.96	-2.10	0.75	0.41	0.76	0.94	-2.60	0.77	0.54	0.76	0.98	-1.28	0.95
	GAN	0.00	0.41	0.82	-5.95	0.32	0.00	0.41	0.80	-6.24	0.18	0.00	0.84	0.82	-6.42	0.11
	SCPN	0.27	0.69	0.89	-3.48	0.63	0.19	0.49	0.90	-3.30	0.63	0.41	0.80	0.90	-3.23	0.80
	XVD (mean)	0.28	0.89	0.85	-3.52	0.77	0.25	0.97	0.86	-3.95	0.72	**0.87**	0.97	0.92	-2.64	0.95
	XVD (std)	*0.10*	*0.02*	*0.02*	*0.23*	*0.03*	*0.15*	*0.04*	*0.01*	*0.21*	*0.21*	*0.00*	*0.00*	*0.00*	*0.02*	*0.01*

Table 2: Evaluation of baselines and our approach, XVD, across six datasets and two victim models. For XVD, we report the mean and std of each metric across three random seeds (the other approaches are deterministic). We use the following abbreviations: VSR is Validated Success Rate, Flip is the proportion of label flips, Sim is the similarity as measured by BERTScore F1, Flu is fluency as measured by BARTScore, and Ent is the entailment probability measured by an NLI model. Higher is better for all metrics. For dataset abbreviations, see Section 4.1.

$n = 1$. Our findings, depicted as a plot in Figure 2, have confirmed the expected increase in label-flipping rate with larger n. The fluency and similarity metrics have peaked around $n = 4$ before declining, while entailment has remained relatively constant from $n = 8$ onwards. The validated success rate, which compounds the label-flipping rate and the text quality metrics, has improved as n increased, up to a plateau at $n = 32$.

6.2 KL divergence and diversity coefficients

The KL divergence and diversity coefficients (respectively, α_{KL} in Equation 1 and α_d in Equation 3) define the intensity of their respective regularisers and substantially impact the quality and diversity of the generated text, as shown in Figure 3. Increasing the KL coefficient ties the trained model more strongly to the reference model, which in our implementation increases the attack quality at the expense of the label- flipping rate. On the other hand, lower values of the diversity coefficient push the model towards samples that are less diverse, while higher values promote diversity per se. Empirically, we have found that that the label-flipping rate has tended to remain constant for a range of diversity values, but the overall text quality metrics have peaked for a value of 10.

6.3 Impact of the vocabulary mapping

To probe the impact on performance of the vocabulary mapping, we have also carried out an experiment attacking a T5 victim model, which has the same vocabulary as the generative model and dispenses with the need for a vocabulary-mapping ma-

Dataset			Label
HS	Orig	Get two birds stoned at one time.	Neither
	Adv	At the same time get two birds stoned.	Offensive
TREC	Orig	What is the atomic weight of silver?	Numeric
	Adv	Tell me the atomic weight of silver?	Description
SUBJ	Orig	“ funny valentine “ is about learning what it takes to find true love .	Objective
	Adv	funny valentine’s about finding true love.	Subjective
RT	Orig	suffers from unlikable characters and a self-conscious sense of its own quirky hipness .	Negative
	Adv	it is characterized by characters who are unlikable and it has a sense of hipness that is self-conscious. ”	Positive
FP	Orig	In addition , the company will reduce a maximum of ten jobs.	Negative
	Adv	It has announced it has a maximum of ten job-separation measures. It has announced it has	Neutral
Emotion	Orig	i find myself feeling anxious and unsure	Fear
	Adv	anxiety and a lack of confidence	Joy

Table 3: Successful adversarial attack examples generated by the proposed approach.

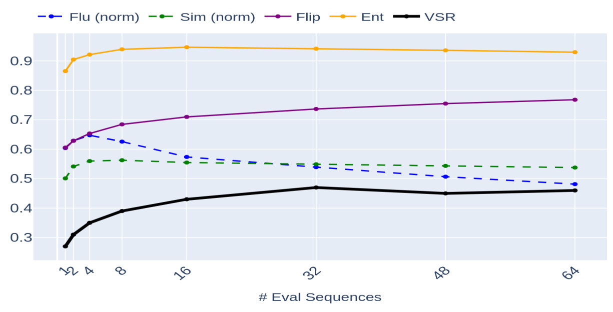

Figure 2: Performance as a function of the number of sequences generated per sample (on the FP dataset). The fluency and similarity metrics have been normalised to the [0,1] interval. Higher is better for all metrics.

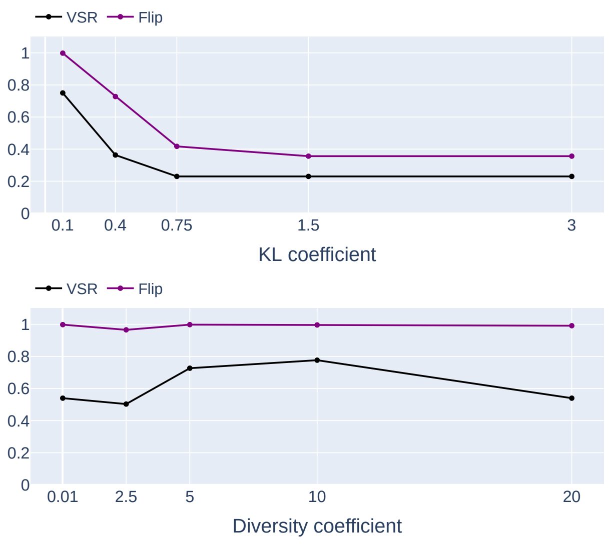

Figure 3: Performance as a function of the KL divergence and diversity coefficients.

trix. The attacks on the T5 model (on the FP dataset, averaged across three seeds) have resulted in a higher VSR value (0.44) compared to ELECTRA (0.39) and DistilBERT (0.28), implying that the vocabulary-mapping matrix may introduce some performance penalisation. However, this difference could also be due to other reasons, such as the homogeneity between the attacker and the victim model. Since it is not obvious how to precisely excise the impact of the vocabulary mapping from that of the other components, we leave a more exact quantification and possible mitigations to future work .

7 Conclusion

This paper has presented an approach for creating white-box adversarial attacks against any text classifier, regardless of the vocabulary of the generator, victim or component models. The approach leverages vocabulary-mapping matrices ro remap the vocabularies across components, allowing for a fully-differentiable training objective without sacrificing the expressiveness of the generator. Experimental results across six datasets and two victim models have confirmed the viability and effectiveness of the proposed approach, and an ablation analysis has shown the impact of the key parameters on the label-flipping/text quality trade-off. Future research might aim to integrate other components, including possibly human preferences, in the training objective, enhance the performance contribution of the vocabulary-mapping matrices, and adapt the approach to tackle other NLP tasks.

8 Ethical considerations

The proposed approach potentially raises two main ethical considerations. The first, is the potential to generate offensive or inappropriate content. However, this risk, largely influenced by the training data and the pretraining of the generative model used, is a common challenge across text generation models and not specifically our work. The second is that the proposed approach might be used by a malicious actor to deceive or manipulate real-world systems. This risk follows from the dual-use nature of adversarial research, where developing methods to defend systems against attacks first requires exploring the attacks themselves.

References

Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Prithiviraj Damodaran. 2021. Parrot: Paraphrase generation for nlu.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515.

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.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. 2021. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5747–5757, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. 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 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 8018–8025.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

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.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 4208–4215. AAAI Press.

P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65.

Paul Michel, Xian Li, Graham Neubig, and Juan Pino. 2019. On evaluation of adversarial perturbations for sequence-to-sequence models. 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 3103–3114.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*.

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

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. 2021. Mind the style of text! adversarial and backdoor attacks based on text style transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4569–4580, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Liwei Song, Xinwei Yu, Hsuan-Tung Peng, and Karthik Narasimhan. 2021. Universal adversarial attacks with natural triggers for text classification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3724–3733, Online. Association for Computational Linguistics.

Ashwin Vijayakumar, Michael Cogswell, Ramprasaath Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *CoRR*, abs/1610.02424.

Eric Wallace, Jens Tuyls, Junlin Wang, Sanjay Subramanian, Matt Gardner, and Sameer Singh. 2019. AllenNLP Interpret: A framework for explaining predictions of NLP models. In *Empirical Methods in Natural Language Processing*.

Tianlu Wang, Xuezhi Wang, Yao Qin, Ben Packer, Kang Li, Jilin Chen, Alex Beutel, and Ed Chi. 2020. CAT-gen: Improving robustness in NLP models via controlled adversarial text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5141–5146, Online. Association for Computational Linguistics.

Catherine Wong. 2017. Dancin seq2seq: Fooling text classifiers with adversarial text example generation.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Ying Xu, Xu Zhong, Antonio Jimeno Yepes, and Jey Han Lau. 2021. Grey-box adversarial attack and defence for sentiment classification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4078–4087, Online. Association for Computational Linguistics.

Jin Yong Yoo, John Morris, Eli Lifland, and Yanjun Qi. 2020. Searching for a search method: Benchmarking search algorithms for generating NLP adversarial examples. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 323–332, Online. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. BARTscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Guoyang Zeng, Fanchao Qi, Qianrui Zhou, Tingji Zhang, Bairu Hou, Yuan Zang, Zhiyuan Liu, and Maosong Sun. 2021. Openattack: An open-source textual adversarial attack toolkit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 363–371.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational

Linguistics, pages 5564–5569, Florence, Italy. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *International Conference on Learning Representations (ICLR)*.

A Training details

The hyperparameters used for our experiments are listed in Table 4. As optimiser, we have used Adafactor[4] with learning rate set to 0.0004. To select the values for the α parameters, we have performed a grid search using the validation set of the FP dataset, selecting the combination that maximised the validation criteria. The β coefficients have been set in a similar way, but with a much smaller search. For the τ parameter of the Gumbel-softmax sampling, we have investigated a range of values (0.25, 0.5, 0.75, 1, 1.25, 1.5), but found no clear performance differences between them, so we have simply set $\tau = 1.25$ for all experiments as it seemed a sensible default. A sensitivity analysis to some of these parameters has been presented in the ablations (Section 6). The model has been trained on a single NVIDIA A40 GPU with 48 GB RAM. We have used the attack success rate over the validation set as the validation criterion, and either early stopped based on a patience parameter, or stopped after a maximum of 12h of training time. We have collated the training batches by bucketing examples with similar length, and then randomising the batches. The details of the various models used — generative and reference model, victim models, and loss component models — are given in Table 5.

Basic statistics from each dataset used are presented in Table 1. For each dataset we have used pre-defined train/val/test splits if available, and otherwise constructed them by randomly selecting 10% of the data as the validation set and 10% as the test set (done for TREC, FP, SUBJ, and HS). For the FP dataset, we have used the dataset version with at least 50% annotator label agreement. For all datasets, we have excluded the training examples that the victim model classified incorrectly, as they could be said to be already "adversarial". We have also only included examples with 32 tokens or fewer, since the pretrained paraphrase model was trained for sequences in that range.

Hyperparameter	**Value**
General	
Optimisation algorithm	Adafactor
Learning rate	4×10^{-5}
Weight decay	0
Batch size (train)	12
Max original length	32
Min generated length	$\max(0, l-2-\text{floor}(l/4))$
Max generated length	$l+2$
Validation frequency	Every 24 batches
Patience	35
Precision	fp32
Coefficients (Equation 3)	
Victim (α_v)	20
Similarity (α_s)	3.5
Entailment (α_e)	0.5
KL (α_{KL})	0.1
Diversity (α_d)	10
Victim threshold (β_e)	$1 - 1/c$
Similarity threshold (β_v)	0.3
Entailment threshold (β_e)	0.4
KL threshold (β_v)	4.5
Gumbel-softmax temperature τ	1.25
# Gumbel samples	5
Test-set generation	
Batch size (eval)	8
# generated sequences (n)	32
# beams	32
# beam groups	16
Diversity penalty	1
Top-p	0.98
Temperature	1

Table 4: Hyperparameters. l is the batch length of generated text during evaluation (after padding) and c is the number of classes in the dataset

B Token mapping rules

In our implementation, the vocabulary of the generative paraphrase model (of size 32,100) is constructed using the SentencePiece (Kudo and Richardson, 2018) tokenisation algorithm, while the component models' vocabulary (of size 30,522) use the WordPiece (Wu et al., 2016) tokenisation algorithm. These algorithms have considerable differences, and mapping the tokens is not straightforward, but we have been able to construct a workable mapping with the following rules.

1. Map special tokens (e.g., PAD, EOS, UNK) directly across both vocabularies. Map the extra id tokens in the T5 vocabulary to the UNK WordPiece token. Matches: 104

2. Map one-to-one direct matches between SentencePiece start-of-word tokens and WordPiece non-continuation tokens. Matches: around 9000.

3. Map one-to-one direct matches between SentencePiece non start-of-word tokens and WordPiece continuation tokens. Matches: around 2000.

4. Map remaining SentencePiece tokens one-to-

[4]We set the following arguments: scale_parameter=False, relative_step=False, warmup_init=False)

Purpose	**Size (MB)**	**Identifier**
Training		
Generative model	892	prithivida/parrot_paraphraser_on_T5 (Damodaran, 2021)
Reference model	892	prithivida/parrot_paraphraser_on_T5 (Damodaran, 2021)
Victim model (ELECTRA)	54	google/electra-small-discriminator
Victim model (DistilBERT)	268	distilbert-base-uncased
Similarity	134	sentence-transformers/paraphrase-MiniLM-L12-v2
Entailment	54	howey/electra-small-mnlis
Evaluation		
Similarity (BERTScore)	1630	roberta-large
Fluency (BARTScore)	1630	facebook/bart-large-cnn
Entailment	54	howey/electra-small-mnli

Table 5: The models used in this paper. Our GPU memory requirements dictated we use only small, distilled models, but it is highly likely larger models would give better performance. All models are from the Hugging Face Model Hub.

many with WordPiece tokens using the WordPiece tokeniser, stripping any generated special tokens, and assigning equal probabilities to all matches. The few tokens without matches (in practice, special cases like \xad) are mapped to the UNK token. Matches: around 22k.

C Limitations

A limitation of the proposed approach is its substantial requirement for computing and hardware resources, as gradients need to be computed across multiple models. A GPU with at least 24 GB of memory is likely the minimum computing requirement, and for the utilization of larger models, correspondingly more memory is necessary. This limitation will lessen in the future as GPUs with larger memory will become routinely available, and as advancements in model distillation techniques continue. Another limitation is that the approach has been solely tested with a paraphrase model as the generator, and over datasets comprised of relatively short sentences. The generalizability of the method to other models and text styles remains then an open question. However, the possibility of subdividing larger blocks of text suggests that this may not be a significant limitation in practice.