Generating Natural Adversarial Examples

Zhengli Zhao, Dheeru Dua & Sameer Singh
Department of Computer Science
University of California
Irvine, CA 92697, USA
{zhengliz,ddua,sameer}@uci.edu

Abstract

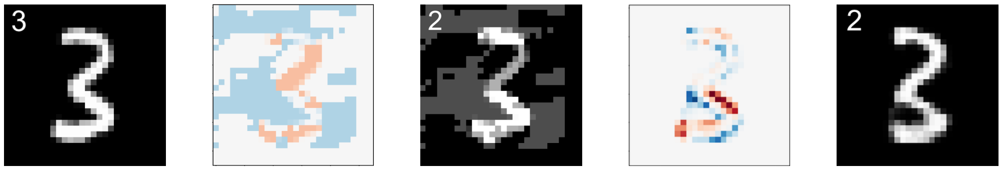
Due to their complex nature, it is hard to characterize the ways in which machine learning models can misbehave or be exploited when deployed. Recent work on adversarial examples, i.e. inputs with minor perturbations that result in substantially different model predictions, is helpful in evaluating the robustness of these models by exposing the adversarial scenarios where they fail. However, these malicious perturbations are often unnatural, not semantically meaningful, and not applicable to complicated domains such as language. In this paper, we propose a framework to generate natural and legible adversarial examples by searching in semantic space of dense and continuous data representation, utilizing the recent advances in generative adversarial networks. We present generated adversaries to demonstrate the potential of the proposed approach for black-box classifiers in a wide range of applications such as image classification, textual entailment, and machine translation. We include experiments to show that the generated adversaries are natural, legible to humans, and useful in evaluating and analyzing black-box classifiers.

1 Introduction

With the impressive success and extensive use of machine learning models in various security-sensitive applications, it has become crucial to study vulnerabilities in these systems. Dalvi et al. (2004) show that adversarial manipulations of input data often result in incorrect predictions from classifiers. This raises serious concerns regarding the security and integrity of existing machine learning algorithms, especially when even state-of-the-art models including deep neural networks have be shown to be highly vulnerable to adversarial attacks with intentionally worst-case perturbations to the input (Szegedy et al., 2014; Goodfellow et al., 2015; Kurakin et al., 2016; Papernot et al., 2016a; Kurakin et al., 2017). These adversaries are generated effectively with access to the gradients of target models, resulting in much higher successful attack rates than data perturbed by random noise of even larger magnitude. Further, training models by including such adversaries can provide machine learning models with additional regularization benefits (Goodfellow et al., 2015).

Although these adversarial examples expose "blind spots" in machine learning models, they are *unnatural*, i.e. these worst-case perturbed instances are not ones the classifier is likely to face when deployed. Due to this, it is difficult to gain helpful insights into the fundamental decision behavior inside the black-box classifier: why is the decision different for the adversary, what can we change in order to prevent this behavior, is the classifier robust to natural variations in the data when not in an adversarial scenario? Moreover, there is often a mismatch between the input space and the *semantic space* that we can understand. Changes to the input we may not think meaningful, like slight rotation or translation in images, often lead to substantial differences in the input instance. For example, Pei et al. (2017) show that minimal changes in the lighting conditions can fool automated-driving systems, a behavior adversarial examples are unable to discover. Due to the unnatural perturbations, these approaches cannot be applied to complex domains such as language, in which enforcing grammar and semantic similarity is difficult when perturbing instances. Therefore, existing approaches that find adversarial examples for text often result in ungrammatical sentences, as in the examples generated by Li et al. (2016a), or require manual intervention, as in Jia & Liang (2017a).

In this paper, we propose a framework for generating *natural* adversarial examples, i.e. instances that are meaningfully similar, valid/legible, and helpful for interpretation. The primary intuition



(a) Input instance (b) FGSM Δx (c) FGSM adversary (d) Our Δx (e) Our adversary "2"

Figure 1: **Adversarial examples.** Given an instance (a), existing FGSM approach (Goodfellow et al., 2015) adds small perturbations in (b), that change the prediction of the model (to be "2", in this case). Instead of such random-looking noise, our framework generates natural adversarial examples, such as in (e), where the differences, shown in (d) (with blue/+, red/-), are meaningful changes to the strokes.

behind our proposed approach is to perform the search for adversaries in a dense and continuous representation of the data instead of searching in the input data space directly. We employ generative adversarial networks (GANs) (Goodfellow et al., 2014) to learn a projection to map normally distributed fixed-length vectors to data instances. Given an input instance, we search for adversaries in the neighborhood of its corresponding representation in latent space by sampling within a range that is iteratively incremented. Figure 1 provides an example of adversaries for digit recognition. Given a multi-layer perceptron (MLP) for MNIST and an image from test data (Figure 1a), our approach generates a *natural* adversarial example (Figure 1e) which is classified incorrectly as "2" by the classifier. Compared to the adversary generated by the existing Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015) that adds gradient-based noise (Figures 1c and 1b), our adversary (Figure 1e) looks like a hand-written digit similar to the original input. Further, the difference (Figure 1d) provides some insights into the behavior of the classifier, such as the fact that slightly thickening (color blue) the bottom stroke of the input and thinning (color red) the one above it, fools the classifier.

We apply our approach to both image and text domains, and generate adversaries that are more natural and grammatical, semantically close to the input, and helpful to interpret the local behavior of black-box models. We present examples of natural adversaries for image classification, textual entailment, and machine translation. Experiments and human evaluations also demonstrate that our approach can help evaluate the *robustness* of black-box classifiers even without labeled training data.

2 Framework for Generating Natural Adversaries

In this section, we describe the problem setup and details of our framework for generating natural adversarial examples of both continuous images and discrete text data. Given a black-box classifier f and a corpus of unlabeled data X, the goal here is to generate adversarial example x^* for a given data instance x that results in a different prediction, i.e. $f(x^*) \neq f(x)$. In general, the instance x is not in X, but comes from the same underlying distribution $\mathcal{P}_x$, which is the distribution we want to generate x^* from as well. We want x^* to be the nearest such instance to x in terms of the low-dimensional manifold that defines the distribution $\mathcal{P}_x$, instead of in the original data representation.

Unlike other existing approaches that search directly in input x space for adversaries, we propose to search in the corresponding dense representation of z space. In other words, instead of finding the adversarial x^* directly, we find the adversarial z^* in the underlying dense vector space which defines the distribution $\mathcal{P}_x$, and then map it back to x^* with the help of generative models. By searching samples in the latent low-dimensional z space and mapping them to x space to identify the adversaries, we encourage these adversaries to be valid (legible for images, and grammatical for sentences) and semantically close to the original input.

Background: Generative Adversarial Networks To tackle the problem described above, we need powerful generative models to learn a mapping from the latent low-dimensional representation to the distribution $\mathcal{P}_x$, which we estimate using samples in X. GANs are a class of such generative models that can be trained via procedures of minimax game between two competing networks (Goodfellow et al., 2014): given a large amount of unlabeled instances X as training data, the generator $\mathcal{G}_\theta$ learns to map some noise with distribution $p_z(z)$ where $z \in \mathbb{R}^d$ to synthetic data that is as close to the training data as possible; on the other hand, the critic $\mathcal{C}_\omega$ is trained to discriminate the output of the

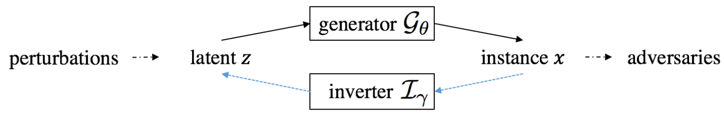


Figure 2: **Model architecture.** Our model consists of an adversarially trained generator, used to decode perturbations of z to query the classifier, and a matching inverter to encode x to z.

generator from real data samples from X. The original objective function of GANs has been found to be hard to optimize in practice, for reasons theoretically investigated in Arjovsky & Bottou (2017). Arjovsky et al. (2017) refine the objective with Wasserstein-1 distance as:

$$\min_\theta \max_\omega \mathbb{E}_{x\sim p_{\text{real}}(x)}[\mathcal{C}_\omega(x)] - \mathbb{E}_{z\sim p_z(z)}[\mathcal{C}_\omega(\mathcal{G}_\theta(z))]. \tag{1}$$

Wasserstein GAN achieves improvement in the stability of learning and provides useful learning curves. A number of further improvements to the GAN framework have been introduced (Salimans et al., 2016; Arjovsky & Bottou, 2017; Gulrajani et al., 2017; Zhao et al., 2017) that we discuss in Section 6. We incorporate the structure of WGAN and relevant improvements as a part of our framework for generating natural examples close to the training data distribution, as we describe next.

Natural Adversaries In order to represent the natural instances of the domain, we first train a WGAN on corpus X, which provides a *generator* $\mathcal{G}_\theta$ that maps random dense vectors $z \in \mathbb{R}^d$ to samples x from the domain of X. We separately train a matching *inverter* $\mathcal{I}_\gamma$ to map data instances to their corresponding dense representations by minimizing the reconstruction error:

$$\min_\gamma \mathbb{E}_{x\sim p_{\text{real}}(x)}\|\mathcal{G}_\theta(\mathcal{I}_\gamma(x)) - x\| + \lambda \mathbb{E}_{z\sim p_z(z)}\|\mathcal{I}_\gamma(\mathcal{G}_\theta(z)) - z\|. \tag{2}$$

Using these learned functions, we define the *natural adversarial example* x^* as the following:

$$x^* = \mathcal{G}_\theta(z^*) \text{ where } z^* = \operatorname*{argmin}_{\tilde{z}} \|\tilde{z} - \mathcal{I}_\gamma(x)\| \text{ s.t. } f(\mathcal{G}_\theta(\tilde{z})) \neq f(x). \tag{3}$$

We perturb the dense representation of x, $z = \mathcal{I}_\gamma(x)$ instead of x itself. We use the generator to test whether a perturbation $\tilde{z}$ fools the classifier f, i.e. $\tilde{x} = \mathcal{G}_\theta(\tilde{z})$ is sent to f. We include a synthetic example for further intuition in the appendix. Figure 2 shows the general architecture of our approach.

Search Algorithm In our implementation (pseudocode in the appendix), we utilize the inverter to obtain the latent vector $z = \mathcal{I}_\gamma(x)$ of x, and feed perturbations $\tilde{z}$ in the neighborhood of z to the generator to generate natural samples $\tilde{x} = \mathcal{G}_\theta(\tilde{z})$. In order to identify the nearest natural sample that changes the prediction from f, we incrementally increase the search range (by Δr) within which the perturbations $\tilde{z}$ are randomly sampled (N samples for each iteration), until we have generated samples x' that change the prediction. Among these samples x', we choose the one which has the closest z^* to the original z as an adversarial example x^*. Our current iterative search algorithm is sample-based and applicable to black-box classifiers with no need of access to their gradients. Further, it is always guaranteed to find an adversary, i.e. one that upper bounds the optimal adversary. Although it is inefficient compared to gradient-based methods, the search is in the compact z-space instead of the high-dimensional x. We are interested in exploring more efficient search methods later.

3 Illustrative Examples

We demonstrate the potential of our approach in generating informative, legible, and natural adversarise by applying it to a number of classifiers for both visual and textual domains.

3.1 Generating Image Adversaries

Image classification has been a focus for adversarial example generation due to the recent successes in computer vision. We apply our approach to two standard datasets, MNIST and LSUN, and present generated natural adversaries. Here we use $\Delta r = 0.01$ and $N = 5000$ as the hyper-parameters.

Handwritten Digits Scans of human-written text provide an intuitive definition of what is *natural*, i.e. do the generated images look like something a person would write? In other words, how would

Table 1: **Adversarial examples of MNIST.** The top row shows images from original test data, and the others show corresponding adversaries generated by FGSM against LeNet and our approach against both RF and LeNet. Predictions from the classifier are shown in the corner of each image.

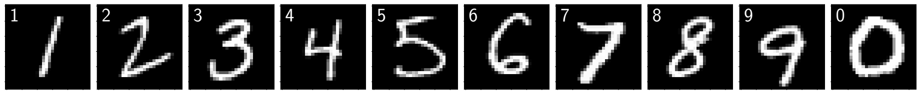
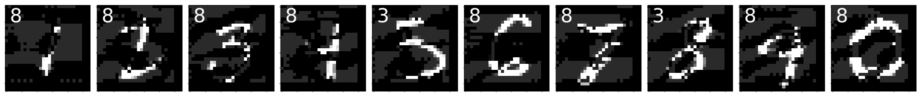
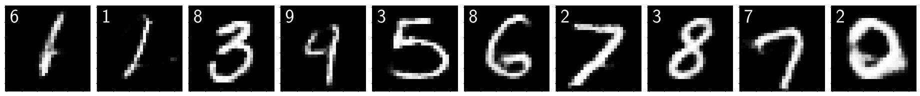
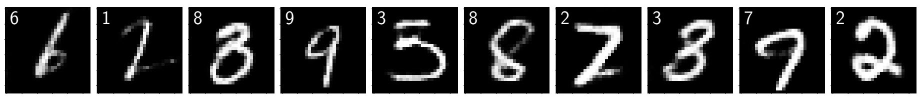
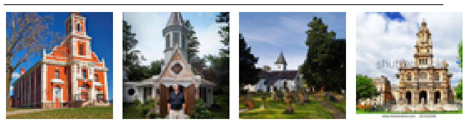
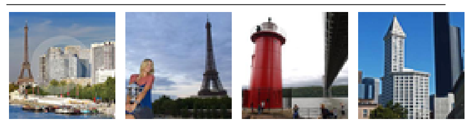
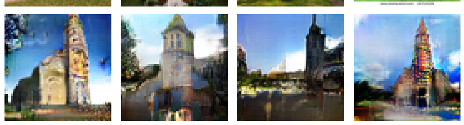
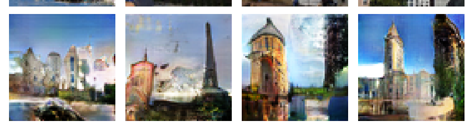
Original	
FGSM (LeNet)	
Random Forests	
LeNet	

Table 2: **Adversarial examples against MLP classifier of LSUN by our approach.** 4 original images each of "Church" and "Tower", with their adversaries of the flipped class in the bottom row.

	church→tower	tower→church
Origin		
Adversary		

a human change a digit in order to fool a classifier. We train a WGAN on 60,000 training images from MNIST following similar procedures as in Gulrajani et al. (2017), with the generator consisting of transposed convolutional layers and the critic consisting of convolutional layers. We include the inverter with fully connected layers on top of critic's last hidden layer. We train two target classifiers to generate adversaries against: Random Forests (RF) with 5 trees (test accuracy 90.45%), and LeNet, as trained in LeCun et al. (1998) (test accuracy 98.71%). We treat both these classifiers as black-boxes, and present the generated adversaries in Table 1 with examples of each digit (from test instances that the GAN or classifiers never observed). Adversaries generated by FGSM look like the original digits eroded by uninterpretable noise (these may not be representative of the approach, as changing ϵ for the method results in substantially different results). Our *natural* adversaries against both classifiers are quite similar to the original inputs in overall style and shape, yet provide informative insights into classifiers' decision behavior around the input. Take the digit "5" as an example: dimming the vertical stroke can fool LeNet into predicting "3". Further we observe that adversaries against RF often look closer to the original images in overall shape than those against LeNet. It implies that compared to LeNet, RF can be fooled by smaller changes to the inputs; in other words, RF is less robust than LeNet in classification. We will return to this observation later.

Church vs Tower We apply our approach to outdoor, color images of higher resolution. We choose the category of "Church Outdoor" in LSUN dataset (Yu et al., 2015), randomly sample the same amount of 126,227 images from the category of "Tower", and resize them to resolution of 64×64. The training procedure is similar to MNIST, except that the generator and critic in WGAN are deep residual networks (He et al., 2016). We train an MLP classifier on these two classes with test accuracy of 71.3%. Table 2 presents original images for both classes and corresponding adversarial examples. From looking at these pairs, we can observe that the generated adversaries make changes that are natural for this domain. For example, to change the classifier's prediction from "Church" to "Tower", the adversaries sharpen the roof, narrow the buildings, or change a tree into a tower. We can observe

Table 3: **Textual Entailment.** For a pair of premise ($\mathbf{p}:$) and hypothesis ($\mathbf{h}:$), we present the generated adversaries for three classifiers by perturbing the hypothesis ($\mathbf{h}':$). The last column provides the true label, followed by the changes in the prediction for each classifier.

Classifiers	**Sentences**	**Label**
Original	$\mathbf{p}$: The man wearing blue jean shorts is grilling. $\mathbf{h}$: The man is walking his dog.	Contradiction
Embedding	$\mathbf{h}'$: The man is walking by the dog .	Contradiction $\rightarrow$ Entailment
LSTM	$\mathbf{h}'$: The person is walking a dog .	Contradiction $\rightarrow$ Entailment
TreeLSTM	$\mathbf{h}'$: A man is winning a race .	Contradiction $\rightarrow$ Neutral

similar behavior in the other direction: the image with the Eiffel Tower is changed to a "church" by converting a woman into a building, and narrowing the tower.

3.2 Generating Text Adversaries

Generating grammatical and linguistically coherent adversarial sentences is a challenging task due to the discrete nature of text: adding *imperceptible* noise is impossible, and most actual changes to x may not result in grammatical text. Prior work on generating textual adversaries (Li et al., 2016b; Alvarez-Melis & Jaakkola, 2017; Jia & Liang, 2017b) performs word erasures and replacements directly on text input space x, using domain-specific rule based or heuristic based approaches, or requires manual intervention. Our approach, on the other hand, performs perturbations in the continuous space z, that has been trained to produce semantically and syntactically coherent sentences automatically.

We use the adversarially regularized autoencoder (ARAE) (Zhao et al., 2017) for encoding discrete text into continuous codes. ARAE model encodes a sentence with an LSTM encoder into continuous code and then performs adversarial training on these codes to capture the data distribution. We introduce an inverter that maps these continuous code representations into the Gaussian space of z. We use a four-layer strided CNN for the encoder as it yields more coherent sentences than LSTMs from the ARAE model, however LSTM works well as the decoder. We train two MLP models for the generator and the inverter, to learn mappings between noise and continuous codes. We train our framework on the SNLI (Bowman et al., 2015) data with the same preprocessing as Zhao et al. (2017), use hyper-parameters $\Delta r = 0.01$ and $N = 100$, and present sample perturbations in the appendix.

Textual Entailment Textual Entailment (TE) is a task designed to evaluate common-sense reasoning for language, requiring both natural language understanding and logical inferences for text snippets. In this task, we classify a pair of sentences, a *premise* and a *hypothesis*, into three categories depending on whether the hypothesis is *entailed* by the premise, *contradicts* the premise, or is *neutral* to it. For instance, the sentence "There are children present" is entailed by the sentence "Children smiling and waving at camera", while the sentence "The kids are frowning" contradicts it. We use our approach to generate adversaries by perturbing the hypothesis to deceive classifiers, keeping the premise unchanged. We train three classifiers of varying complexity, namely, an *embedding* classifier that is a single layer on top of the average word embeddings, an *LSTM* based model consisting of a single layer on top of the sentence representations, and *TreeLSTM* (Chen et al., 2017) that uses a hierarchical LSTM on the parses and is a top-performing classifier for this task. A few examples comparing the three classifiers are shown in Table 3 (more examples in the appendix). Although all classifiers correctly predict the label, as the classifiers get more accurate (from *embdedding* to *LSTM* to *TreeLSTM*), they require much more substantial changes to the sentences to be fooled.

Machine translation We consider machine translation because not only is it one of the most successful applications of neural approaches to NLP, but also most practical translation system lie behind black-box access APIs. The notion of *adversary*, however, is not so clear here as the output of a translation system is not a class. Instead, we define adversary for machine translation relative to a *probing function* that tests the translation for certain properties, ones that may lead to linguistic insights into the languages, or detect potential vulnerabilities. We use the same generator and inverter as in entailment, and find such "adversaries" via API access to the currently deployed Google Translate model (as of *October 15, 2017*) from English to German.

Table 4: **Machine Translation.** “Adversary” that introduces the word “stehen” into the translation.

Source Sentence (English)	Generated Translation (German)
$\mathbf{s}$: A man and woman **sitting** on the sidewalk .	Ein Mann und eine Frau, die auf dem Bürgersteig **sitzen**.
$\mathbf{s}'$: A man and woman **stand** on the bench .	Ein Mann und eine Frau **stehen** auf der Bank.

Table 5: **“Adversaries” to find dropped verbs.** The left column contains the original sentence s and its adversary s', while the right contains their translations, with English translation in red.

Source Sentence (English)	Generated Translation (German)
$\mathbf{s}$: People sitting in a dim restaurant **eating**	Leute, die in einem dim Restaurant **essen** sitzen.
$\mathbf{s}'$: People sitting in a living room **eating** .	Leute, die in einem Wohnzimmeressen sitzen. *(People sitting in a living room)*
$\mathbf{s}$: Elderly people **walking** down a city street .	Ältere Menschen, die eine Stadtstraße **hinuntergehen**.
$\mathbf{s}'$: A man **walking** down a street playing	Ein Mann, der eine Straße entlang spielt. *(A man playing along a street.)*

First, let us consider the scenario in which we want to generate adversarial English sentences such that a specific German word is introduced into the German translation. The probing function here would test the translation for the presence of that word, and we would have found an adversary (an English sentence) if the probing function *passes* for a translation. We provide an example of such a probing function that introduces the word “stehen” (“stand” in English) to the translation in Table 4 (more examples in the appendix). Since the translation system is quite strong, such adversaries are not surfacing the vulnerabilities of the model, but instead can be used as a tool to understand or learn different languages (in this example, help a German speaker learn English).

We can design more complex probing functions as well, especially ones that target specific vulnerabilities of the translation system. Let us consider translations of English sentences that contain two active verbs, e.g. “People sitting in a restaurant eating”, and see that the German translation has the two verbs as well, “essen” and “sitzen”, respectively. We now define a probing function that passes only if the perturbed English sentence s' contains both the verbs, but the translation only has one of them. An adversary for such a probing function will be an English sentence (s') that is similar to the original sentence (s), but for some reason, its translation is missing one of the verbs. Table 5 presents examples of generated adversaries using such a probing function (with more in the appendix). For example, one that tests whether “essen” is dropped from the translation when its English counterpart “eating” appears in the source sentence (“People sitting in a living room eating.”). These adversaries thus suggest a vulnerability in Google’s English to German translation system: a word acting as a gerund in English often gets dropped from the translation.

4 Experiments

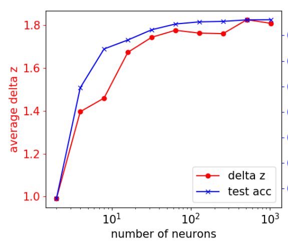
In this section, we demonstrate that our approach can be utilized to compare and evaluate the *robustness* of black-box models even without labeled data. We present experimental results on images and text data with evaluations from both statistical analysis and pilot user studies.

Robustness of Black-box Classifiers We apply our framework to various black-box classifiers for both images and text, and observe that it is useful for evaluating and interpreting these models via comparisons. The primary intuition behind this analysis is that more accurate classifiers often require more substantial changes to the instance to change their predictions, as noted in the previous section.

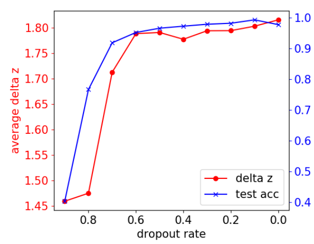
In order to quantify the extent of change for an adversary, the change in the original x representation may not be meaningful, such as RMSE of the pixels or string edit distances, for the same reason we are generating *natural* adversaries: they do not correspond to the *semantic* distance underlying the data manifold. Instead we use the distance of the adversary in the latent space, i.e. $\Delta z = \|z^* - z\|$, in order to measure how much each adversary is modified to change the classifier prediction. We also consider the set of adversaries generated for each instance against a group of classifiers, and count how many times the adversary of each classifier has the highest Δz. We present these statistics in Table 6 for both MNIST (over 100 test images, 10 per digit) and Textual Entailment (over 1260 test

Table 6: **Statistics of adversaries against models for both MNIST and TE.** We include the average Δz for the adversaries and the proportion where each classifier's adversary has the largest Δz compared to the others for the same instance (significant with $p < 0.0005$ using the sign test). The higher values correspond to stronger robustness, as is demonstrated by higher test accuracy.

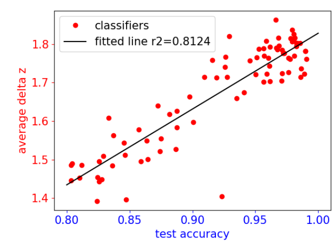
		Average Δz	P(largest Δz)	Test accuracy (%)
MNIST	Random Forests	1.37	0.32	90.45
	LeNet	1.75	0.68	98.71
Entailment	Embeddings	0.12	0.11	62.04
	LSTM	0.14	0.35	69.60
	TreeLSTM	0.17	0.54	89.04



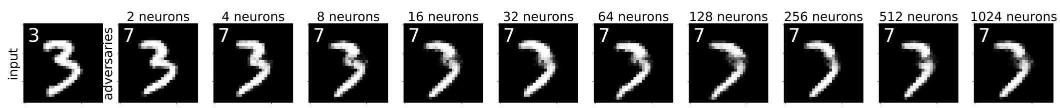
(a) Varying number of neurons



(b) Varying dropout rate



(c) Variants of classifiers



(d) Adversaries for the same input digit against 10 classifiers with increasing capacity.

Figure 3: **Classifier accuracy and average Δz of their adversaries.** In (a) and (b) we vary the number of neurons and dropout rate, respectively. In (c) we present the correlation between accuracy and average Δz for 80 different classifiers. (d) shows adversaries for an input image, against a set of classifiers with a single hidden layer, but varying number of neurons.

sentences), against the classifiers we described in Section 3. For both the tasks, we observe that more accurate classifiers require larger changes to the inputs (by both measures), indicating that generating such adversaries, even for unlabeled data, can evaluate the accuracy of black-box classifiers.

We now consider evaluation on a broader set of classifiers, and study the effect of changing hyper-parameters of models on the results (focusing on MNIST). We train a set of neural networks with one hidden layer by varying the number of neurons exponentially from 2 to 1024. In Figure 3a, we observe that the average Δz of advesaries against these models has a similar trend as their test accuracy. The generated adversaries for a single digit "3" in Figure 3d verify this observation: the adversaries become increasingly different from the original input as classifiers become more complex. We provide similar analysis by fixing the model structure but varying the dropout rates from 0.9 to 0.0 in Figure 3b, and observe a similar trend. To confirm that this correlation holds generally, we train 80 total classifiers that differ in the layer sizes, regularization, and amount of training data, and plot their test set accuracy against the average magnitude of change in their adversaries in Figure 3c. Given this strong correlation, we are confident that our framework for generating natural adversaries can be useful for automatically evaluating black-box classifiers, even in the absence of labeled data.

Table 7: **Pilot study with MNIST**

	RF	LeNet
Looks handwritten?	0.88	0.71
Which closer to original?	0.87	0.13
Agree with the classifier?	0.17	0.59

Table 8: **Pilot study with Entailment**

	LSTM	TreeLSTM
Is adversary grammatical?	0.86	0.78
Is it similar to the original?	0.81	0.58
Agree with the classifier?	0.37	0.58

Human Evaluation We carry out a pilot study with human subjects to evaluate how natural the generated adversaries are, and whether the adversaries they think are similar to the original ones correspond with the less accurate classifiers (as in the evaluation presented above). For both image classification and textual entailment, we select a number of instances randomly, generate adversaries for each against two classifiers, and present a questionnaire to the subjects that evaluates: (1) how natural or legible each generated adversary is; (2) which of the two adversaries is closer to the original image; and (3) what label they would assign to the adversary.

For hand-written digits from MNIST, we pick 20 images (2 for each digit), generate adversaries against RF and LeNet (two adversaries for each image), and obtain 13 responses for each of the questions. In Table 7, we see that the subjects agree that our generated adversaries are quite natural, and also, they find RF adversaries to be much closer to the original image than LeNet (i.e. more accurate classifiers, as per test accuracy on their provided labels, have more distant adversaries). We carry out a similar pilot study for the textual entailment task to evaluate the quality of the perturbed sentences. We present a set of 20 pairs of sentences (premise and hypothesis), and adversarial hypotheses against both LSTM and TreeLSTM classifiers, and receive 4 responses for each of the questions above. The results in Table 8 also validate our previous results: the generated sentences are found to be grammatical and legible, and classifiers that need more substantial changes to the hypothesis tend to be more accurate. We leave a more detailed user study for future work.

5 Related Work

The fast gradient sign method (FGSM) has been proposed in Goodfellow et al. (2015) to generate adversarial examples *fast* rather than optimally. Intuitively, the method shifts the input by ϵ in the direction of minimizing the cost function. Kurakin et al. (2016) propose a simple extension of FGSM by applying it multiple times, which generates adversarial examples with higher successful attack rate, but the underlying idea is the same. Another method known as the Jacobian-based saliency map attack (JSMA) has been introduced by Papernot et al. (2016a). Unlike FGSM, JSMA generates adversaries by greedily modifying the input instance feature-wise. A saliency map is computed with gradients to indicate how important each feature is for the prediction, and the most important one is modified repeatedly until the instance changes the resulting classification. Moreover, it has been observed in practice that adversarial examples designed against a model are often likely to successfully attack another model for the same task that has not been given access to. This transferability property of adversarial examples makes it more practical to attack and evaluate deployed machine learning systems in realistic scenarios (Papernot et al., 2016b; 2017).

All these attack methods above are based on gradients with access to the parameters of differentiable classifiers. Moosavi-Dezfooli et al. (2016) try to find a single noise vector which can cause imperceptible changes in most of data points, and at the same time reduce the classifier accuracy significantly. Our method is capable of generating adversaries against black-box classifiers, even those without gradients such as Random Forests. Also, the noise added by these methods is uninterpretable, while the natural adversaries generated by our approach provide informative insights into the decision behavior of classifiers.

Due to the complex discrete nature of text, adversaries for text have received less attention. Jia & Liang (2017a) generate adversarial examples for evaluating reading comprehension systems with predefined rules and candidate words for substitution after analyzing and rephrasing the input sentences. Li et al. (2016a) introduce a framework to understand neural network through different levels of representation erasure. However, erasure of words or phrases directly often harms text integrity, resulting in semantically or grammatically incorrect sentences. With the help of expressive generative models, our approach perturbs the latent coding of sentences, resulting in legible generated sentences that are semantically similar to the original input. These merits make our framework suitable for text applications such as sentiment analysis, textual entailment, and machine translation.

6 Discussion and Future Work

Our framework fundamentally builds upon GANs as the generative models, and thus the capabilities of GANs directly effects the quality of generated examples. In visual domains, although there have been lots of appealing results produced by GANs, the training is well known to be brittle.

Many recent approaches address how to improve the training stability and the objective function of GANs (Salimans et al., 2016; Arjovsky et al., 2017). Gulrajani et al. (2017) further improve the training of WGAN with regularization of gradient penalty instead of weight clipping. In our practice, we observe that we need to carefully balance the capacities of the generator, the critic, and the inverter that we introduced, to avoid situations such as model collapse. For natural languages, because of the discrete nature and non-differentiability, applications related to text generation have been relatively less studied. Zhao et al. (2017) propose to incorporate a discrete structure autoencoder with continuous code space regularized by WGAN for text generation. Given that there are some concerns about whether GANs actually learn the distribution (Arora & Zhang, 2017), it is worth noting that we can also incorporate other generative models such as Variational Auto-Encoders (VAEs) (Kingma & Welling, 2014) into our framework, as used in Hu et al. (2017) to generate text with controllable attributes, whic we will explore in the future. We focus on GANs because adversarial training often results in higher quality images, while VAEs tend to produce blurrier ones (Goodfellow, 2016). Note that as more advanced GANs are introduced to address their issues, they can be directly incorporated into our framework.

Our proposed algorithm for searching in the semantic space for adversaries is computationally expensive since it is based on sampling and local-search. Search based on gradients such as FGSM are not applicable to our setup because of black-box classifiers and discrete domain applications. We can improve the search by using a coarse-to-fine strategy that finds the upper-bounds by using fewer samples, and then performs finer search in the restricted range. The accuracy of our inverter mapping the input to its corresponding dense vector in latent space is also important for searching adversaries in the right neighborhood. In our experiments, we find that fine-tuning the latent vector produced by the inverter with the GAN fixed can further refine the generated adversarial examples, and we will investigate other such extensions of the search in future.

7 Conclusions

In this paper, we propose a framework for generating *natural* adversaries against black-box classifiers, and apply the same approach to both visual and textual domains. We obtain adversaries that are legible, grammatical, and meaningfully similar to the input. We show that these natural adversaries can help in interpreting the decision behavior and evaluating the accuracy of black-box classifiers even in absence of labeled training data. Our approach, built upon recent work in GANs, is applied to generating adversaries for a wide range of applications including image classification, textual entailment, and machine translation (via the Google Translate API). We will publicly release all of the software and datasets to reproduce our results on publication.

References

David Alvarez-Melis and Tommi S Jaakkola. A causal framework for explaining the predictions of black-box sequence-to-sequence models. *arXiv preprint arXiv:1707.01943*, 2017.

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*, 2017. URL `http://arxiv.org/abs/1701.04862`.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Sanjeev Arora and Yi Zhang. Do gans actually learn the distribution? an empirical study. *arXiv preprint arXiv:1706.08224*, 2017.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced lstm for natural language inference. In *Proc. ACL*, 2017.

Nilesh Dalvi, Pedro Domingos, Sumit Sanghai, Deepak Verma, et al. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 99–108. ACM, 2004.

Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017. URL http://arxiv.org/abs/1704.00028.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *International Conference on Machine Learning*, pp. 1587–1596, 2017.

R. Jia and P. Liang. Adversarial examples for evaluating reading comprehension systems. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2017a.

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

Diederik P Kingma and Max Welling. Stochastic gradient vb and the variational auto-encoder. In *Second International Conference on Learning Representations, ICLR*, 2014.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1611.01236.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016a.

Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016b.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *arXiv preprint arXiv:1610.08401*, 2016.

N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, pp. 372–387, March 2016a. doi: 10.1109/EuroSP.2016.36.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016b.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pp. 506–519, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4944-4. doi: 10.1145/3052973.3053009. URL http://doi.acm.org/10.1145/3052973.3053009.

Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *the 26th ACM Symposium on Operating Systems Principles*, 2017.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL http://arxiv.org/abs/1312.6199.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Junbo Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. Adversarially regularized autoencoders for generating discrete structures. *CoRR*, abs/1706.04223, 2017. URL http://arxiv.org/abs/1706.04223.

APPENDIX

A ILLUSTRATION WITH SYNTHETIC DATA

As shown in Figure 4 with a toy example of synthetic data, we can effectively map data instance x to its corresponding latent dense vector z with the help of the inverter via $\mathcal{I}_\gamma(x)$, and then reconstruct x with the help of the generator via $\mathcal{G}_\theta(\mathcal{I}_\gamma(x))$. For a naive classifier with the horizontal line as decision boundary, adversarial examples should be points above the line given input data x in Figure 4c. By searching in corresponding latent space, our approach finds x^* on the left as a $natural$ adversary because it is the closest one in semantic space (along the curve trace) and it does exist in the data distribution. However, the other gradient-based approach may find the x^* right above x as adversarial in input space regardless of the real distribution.

B ALGORITHM

Algorithm 1 shows the pseudocode of the approach of our framework.

C TEXT PERTURBATIONS

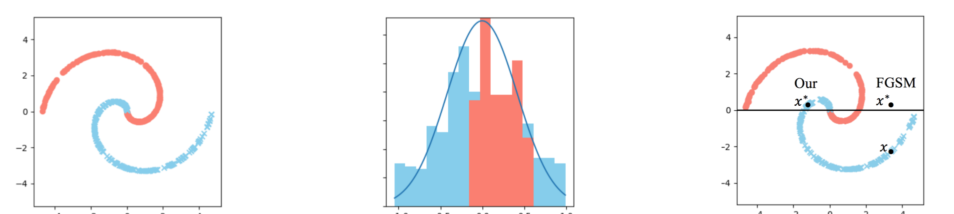
Table 9 shows some examples of the perturbations generated automatically by our approach, which are grammatical and semantically close to the original sentences.

D TEXTUAL ENTAILMENT EXAMPLES

We provide additional examples of generated adversarial hypotheses for sentences from the SNLI corpus in Table 10, which corresponds to the examples in the main text in Table 3.

E MACHINE TRANSLATION EXAMPLES

We provide additional examples of the two probing functions in Table 11 and Table 12, corresponding to Table 4 and Table 5 in the main text, respectively.



(a) Synthetic input data x (b) Latent $z = \mathcal{I}_\gamma(x)$ (c) Reconstructed data $\mathcal{G}_\theta(z)$

Figure 4: **Illustration with synthetic data.** With training data that lies on a complex manifold (a), the inverter maps input to compact *gaussian* latent $z = \mathcal{I}_\gamma(x)$ in (b), while the generator reconstructs the data via $\mathcal{G}_\theta(\mathcal{I}_\gamma(x))$ in (c). Given f as a binary classifier with decision boundary as the horizontal line in (c), for an input x, our approach returns x^* on the left as *natural* adversary that lies on the manifold, while existing approaches may find the right x^* as the adversary, which is the nearest but impossible.

Algorithm 1 Search in latent z space for adversaries

Require: a target black-box classifier f, an input instance x, and a corpus of relevant data X

Hyper-parameters: Δr: increment of r, N: number of samples in each iteration
Train a generator $\mathcal{G}_\theta$ and an inverter $\mathcal{I}_\gamma$ on X
$y \leftarrow f(x)$, $z \leftarrow \mathcal{I}_\gamma(x)$, radius $r \leftarrow 0$
loop ▷ loop till we find an adversary
    $r \leftarrow r + \Delta r$, $S \leftarrow \{\}$ ▷ increase radius
    for sample N random noise vectors ϵ of elements within $(r, r + \Delta r]$ **do**
        $\tilde{z} \leftarrow z + \epsilon$, $\tilde{x} \leftarrow \mathcal{G}_\theta(\tilde{z})$, $\tilde{y} \leftarrow f(\tilde{x})$ ▷ get classifier prediction for the reconstruction
        $S \leftarrow S \cup \langle \tilde{x}, \tilde{y}, \tilde{z} \rangle$
    if $\exists \langle x', y', z' \rangle \in S$ such that $y' \neq y$ **then** ▷ at least one adversary
        $z^* \leftarrow \text{argmin}_{z' s.t. y' \neq y} \|z' - z\|$ ▷ get the nearest one
        return $x^* \leftarrow \mathcal{G}_\theta(z^*)$

Table 9: **Text perturbations.** Examples are generated by perturbing the origins in semantic space.

Original	Some dogs are running on a deserted beach .	A man playing an electric guitar on stage .
Perturbation	Some dogs are running on a grassy field . Some dogs are walking along a path . Some dogs are running down a hill . A dog is running on a grassy field . A dog is running down a trail .	A man is playing an electric guitar . A man is playing an acoustic guitar . A man is playing an accordion . A man is playing with an electronic device . A man is playing with an elephant .

Table 10: **Textual Entailment.** For a pair of premise (**p** :) and hypothesis (**h** :), we present the generated adversaries for three classifiers by perturbing the hypothesis (**h′** :). The last column provides the true label, followed by the changes in the prediction from each classifier.

Classifiers	**Sentences**	**Label**
Original	**p** : The man walks among the large trees. **h** : The man is lost in the woods.	Neutral
Embedding	**h′** : The man is lost at the woods .	Contradiction → Neutral
LSTM	**h′** : The man is crying in the woods .	Neutral → Contradiction
TreeLSTM	**h′** : The man is lost in a bed .	Neutral → Contradiction

Table 11: **Machine Translation.** “Adversaries” that introduce the word “stehen” into the Google translation system by perturbing the English sentence.

Source Sentence (English)	**Generated Translation (German)**
s : Asian women are **sitting** in a Restraunt.	Asiatische Frauen **sitzen** in einem Restaurant.
s′ : Asian kids are **standing** in a Restraunt.	Asiatische Kinder **stehen** in einem Restaurant.
s : People **sitting** on the floor.	Leute **sitzen** auf dem Boden.
s′ : People **standing** on the field.	Leute, die auf dem Feld **stehen**.

Table 12: **“Adversaries” that find dropped verbs in English-To-German translation.** The left column contains the original sentence s and its adversary s'. The right column contains the translations of s and s', with English translation provided for legibility.

Source Sentence (English)	**Generated Translation (German)**
s : A man looks back while laughing and **walking** .	Ein Mann schaut beim Lachen und **Gehen** zurck.
s′ : A man is laughing **walking** down the ground .	Ein Mann lacht auf dem Boden. *(A man laughs on the floor.)*
s : She is cooking food while wearing a **dress** .	Sie kocht Essen, whrend sie ein **Kleid** trgt.
s′ : She is cooking **dressed** for a wedding .	Sie kocht fr eine Hochzeit. *(She cooks for a wedding)*