
Generating Natural Adversarial Examples

Zhengli Zhao
University of California
Irvine, CA 92697
zhengliz@uci.edu

Dheeru Dua
University of California
Irvine, CA 92697
ddua@uci.edu

Sameer Singh
University of California
Irvine, CA 92697
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.

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. [6] show that adversarial manipulations of input data often result in incorrect predictions from classifiers. This raises serious concerns regarding the security of existing machine learning algorithms, especially when even state-of-the-art models including deep neural networks have been shown to be highly vulnerable to adversarial attacks with intentionally worst-case perturbations to the input [8, 14, 15, 18, 21].

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. [19] 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. [17], or require manual intervention, as in Jia and Liang [12].

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 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) [7] to learn a projection to map normally distributed fixed-length

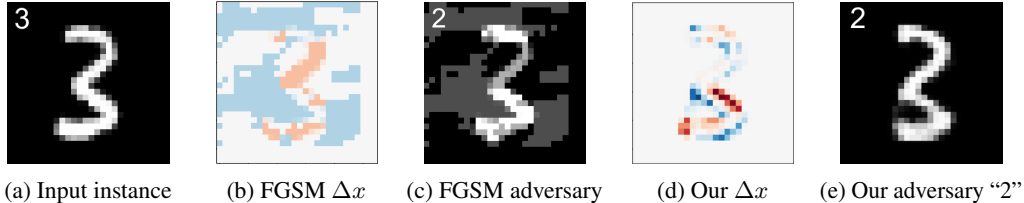


Figure 1: **Adversarial examples.** Given an instance (a), existing FGSM approach [8] adds small perturbations in (b), that change the prediction of the model (to be “2”, in this case). Instead of such random-looking noise, our framework generates natural adversarial examples, such as in (e), where the differences, shown in (d) (with blue/+, red/-), are meaningful changes to the strokes.

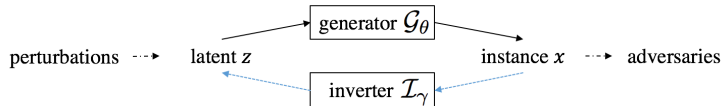


Figure 2: **Model architecture.** Our model consists of an adversarially trained generator, used to decode perturbations of z to query the classifier, and a matching inverter to encode x to z .

vectors to data instances. Given an input instance, we search for adversaries in the neighborhood of its corresponding latent representation by sampling within a range that is iteratively incremented. Fig 1 provides an example of adversaries for digit recognition. Given an MLP classifier for MNIST and an image from test data (Fig 1a), our approach generates a *natural* adversarial example (Fig 1e) which is classified incorrectly as “2” by the classifier. Compared to the adversary generated by the existing Fast Gradient Sign Method (FGSM) [8] that adds gradient-based noise (Fig 1c and 1b), our adversary (Fig 1e) looks like a hand-written digit similar to the original input. Further, the difference (Fig 1d) provides some insights into the behavior of the classifier, such as slightly thickening (color blue) the bottom stroke of the input and thinning (color red) the one above it, fools the classifier.

2 Framework for Generating Natural Adversaries

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. 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. 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.

To tackle the problem described above, we incorporate GANs [7] as powerful generative models to learn how to represent the natural instances of the domain. We first train a Wasserstein GAN [2] on corpus X , which provides a *generator* \mathcal{G}_θ 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}_γ to map data instances to their corresponding dense representations by minimizing the reconstruction error: $\min_\gamma \mathbb{E}_{x \sim p_{\text{real}}(x)} \|\mathcal{G}_\theta(\mathcal{I}_\gamma(x)) - x\| + \lambda \mathbb{E}_{z \sim p_z(z)} \|\mathcal{I}_\gamma(\mathcal{G}_\theta(z)) - z\|$. Using these learned functions, we define the *natural adversarial example* x^* as the following:

$$x^* = \mathcal{G}_\theta(z^*) \text{ where } z^* = \underset{\tilde{z}}{\operatorname{argmin}} \|\tilde{z} - \mathcal{I}_\gamma(x)\| \text{ s.t. } f(\mathcal{G}_\theta(\tilde{z})) \neq f(x). \quad (1)$$

Fig 2 shows the architecture of our approach. In our implementation, 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 within which the perturbations \tilde{z} are randomly sampled, 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

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































Original										
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								

no need of access to their gradients. Although it is inefficient compared to gradient-based methods, it always guarantees to find an adversary, i.e. one that upper bounds the optimal adversary. 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.

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.

Handwritten Digits We train a WGAN on MNIST as in Gulrajani et al. [9], and include an inverter with fully connected layers on top of critic’s last hidden layer. We train two target classifiers: Random Forests (RF) (test accuracy 90.45%), and LeNet as in LeCun et al. [16] (test accuracy 98.71%). As shown in Tab 1, 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. Our approach is helpful to compare and evaluate black-box classifier even in absence of labeled data.

Church vs Tower We apply our approach to outdoor, color images of higher resolution. We choose “Church Outdoor” category in LSUN dataset [22], randomly sample the same amount of 126,227 images from “Tower” category, and resize them to 64×64 . The training procedure is similar to MNIST, except that the generator and critic in WGAN are deep residual networks [10]. Tab 2 presents original images for both classes and corresponding adversaries against an MLP classifier with test accuracy of 71.3%. 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

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

adversaries sharpen the roof or narrow the buildings; and in the other direction, the image with the Eiffel Tower is changed to a “church” by converting a girl into a building and narrowing the tower.

Generating grammatical and linguistically coherent adversarial sentences is a challenging task due to the discrete nature of text . Prior work on generating textual adversaries [1, 12, 17] 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) [23] for encoding discrete text into continuous codes, and introduce an inverter that maps these continuous codes into the Gaussian space of z . Our framework is trained on the SNLI [4] data.

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. 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* [5] that uses a hierarchical LSTM on the parses. A few examples comparing the three classifiers are shown in Table 3. Although all classifiers correctly predict the label, as the classifiers get more accurate (from *embedding* 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 systems 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.

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 Tab 4. 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.

We can also design more complex probing functions, 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 its translation is missing one of the verbs. Table 5 presents examples

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

Source Sentence (English)	Generated Translation (German)
s : A man and woman sitting on the sidewalk .	Ein Mann und eine Frau, die auf dem Bürgersteig sitzen .
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)
s : A man looks back while laughing and walking .	Ein Mann schaut zurück, während er lacht und geht .
s' : A man is laughing walking down the ground .	Ein Mann lacht über den Boden. <i>(A man laughs over the floor.)</i>

of generated adversaries using such a probing function: one that tests whether “essen” is dropped from the translation when its English counterpart “eating” appears in the source (“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 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. Many recent approaches address how to improve the training stability and the objective function of GANs [2, 9, 20]. 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. Given that there are some concerns about whether GANs actually learn the distribution [3], it is worth noting that we can also incorporate other generative models such as Variational Auto-Encoders (VAEs) [13] into our framework, as used in Hu et al. [11] to generate text with controllable attributes, whic we will explore in the future.

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.

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 is applied to generating adversaries for a wide range of applications including image classification, textual entailment, and machine translation.

References

- [1] David Alvarez-Melis and Tommi S. Jaakkola. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2017.
- [2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017.
- [3] Sanjeev Arora and Yi Zhang. Do gans actually learn the distribution? an empirical study. *arXiv preprint 1706.08224*, 2017.
- [4] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2015.
- [5] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced lstm for natural language inference. In *Association for Computational Linguistics (ACL)*, 2017.
- [6] Nilesh Dalvi, Pedro Domingos, Sumitanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.
- [7] 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 (NIPS)*, 2014.
- [8] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.
- [9] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.
- [10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [11] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. Toward controlled generation of text. In *International Conference on Machine Learning (ICML)*, 2017.
- [12] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2017.
- [13] Diederik P. Kingma and Max Welling. Stochastic gradient vb and the variational auto-encoder. In *International Conference on Learning Representations (ICLR)*, 2014.
- [14] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint 1607.02533*, 2016.
- [15] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations (ICLR)*, 2017.
- [16] 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.
- [17] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint 1612.08220*, 2016.
- [18] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy*, 2016.
- [19] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *the ACM Symposium on Operating Systems Principles*, 2017.

- [20] 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 (NIPS)*, 2016.
- [21] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.
- [22] 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 1506.03365*, 2015.
- [23] Junbo Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. Adversarially regularized autoencoders. *arXiv preprint 1706.04223*, 2017.