NeurIPS 2019 Reproducibility Challenge
Adversarial Examples Are Not Bugs, They Are Features

Chengming Hu
School of Computer Science
McGill University
Montreal, QC H3A 0G4
chengming.hu@mail.mcgill.ca

Haolun Wu
School of Computer Science
McGill University
Montreal, QC H3A 0G4
haolun.wu@mail.mcgill.ca

Kai Wang
School of Computer Science
McGill University
Montreal, QC H3A 0G4
kai.wang3@mail.mcgill.ca

Abstract

Adversarial examples have attracted significant attention in machine learning. The reproduced paper proposed that adversarial examples can be directly attributed to the presence of non-robust features: highly predictive, yet brittle features. This challenge aims to reproduce the tasks reported in the paper and modify model components to understand the proposed model's robustness. Hence, we first generated a new robust dataset including adversarial examples, and then reproduced a Residual Neural Network (ResNet) classifier baseline on CIFAR-10 dataset. In addition, some model hyperparameters, such as learning rate, value of adversarial perturbation and normalization approaches, have been changed to explore how these components impact the classification accuracy. We also designed, implemented and evaluated the Visual Geometry Group Network (VGG), DenseNet and InceptionV3 classifiers as the extensions of reproduced paper. It is found that the DenseNet classifier reported the best accuracy of 90.49% in our works, so DenseNet can be recommended in the future CIFAR-10 dataset classification.

1 Introduction

Deep Neural Network (DNN) has been widely used in many areas in this era with data explosion. From relatively simple tasks (face recognition, image classification, object detection, etc.) to more complex tasks(nature language processing, self-driving vehicles, playing chess and goes, etc.), it has achieved an amazing success, to some degree, even outperformed humans. Since a large number of data are involved, a complete feature engineering is less feasible and realistic in practice; this is also why DNN is useful: it does not require too much feature extracting process from humans any more, while an automatic feature extracting can be obtained during its learning procedure. Interestingly, humans can actually merely comprehend a low level of features; conversely, lots of features obtained by deep learning approaches are likely to be imperceptible to humans, which are at a much higher level. For instance, in a simple image classification problem, some quite obvious features like "ears" and "tails" are probably only visible for humans for distinguishing a cat. However, these features may not be that useful for classification; a DNN can decide what features really matter itself.

As deep learning technologies gradually reach state-of-the-art performance, discussion and concerns on security have raised during the past decade, since many applications of DNN are highly connected with data privacy. Recent studies point out that DNN is vulnerable against well-designed input samples. These samples can easily fool a well-performed deep learning model with little perturbations imperceptible to humans [21], which are also well known as adversarial examples.

Reproducibility Challenge for 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

Basically, an adversarial example is an instance with small, intentional feature perturbations that cause a machine learning model to make a false prediction[2], additionally, it is frequently imperceptible for humans comparing to natural inputs.

The reason why people are interested in adversarial examples is because of their wide existence in real world. Previous studies has shown that self-driving cars seem to ignore some important traffic signs if adversarial attacks are implemented; face recognition systems will probably fail to recognize a person because of the adversarial examples; even weapons can be developed by criminals to cheat the scanner scans suitcases in the airport if the machine is vulnerable to adversarial examples. These are not only issues in machine learning technology, but also security problems in society.

Therefore, a comprehensive understanding on adversarial examples are truly necessary and significant for understanding the DNNs more deeply, thus ameliorating the robustness and safety for deep learning approaches. The paper we are going to reproduce is creative to propose a new perspective on the phenomenon of adversarial examples: *Adversarial vulnerability is a direct result of our models' sensitivity to well-generalizing features in the data* [7]. Andrew Ilyas *et al.* prompts to view adversarial examples as a natural consequence of the presence of highly predictive but non-robust features, moreover, a fundamentally human phenomenon [7].

The rest of this paper is organized as follows: Related studies about adversarial examples will be discussed in Section 2. In Section 3, we will explain the works in the reproduced paper, such as the robust features model and the algorithms of finding robust and non-robust features. Section 4 will give the description about CIFAR-10 dataset and our reproduced tasks. The experiment setup and the simulation results are further described in Section 5. Section 6 discusses the conclusion and further improvements of our works.

2 Related Work

The application of adversarial examples has been widely used in prior work, especially the field of adversarial examples generation. Researchers attack their models by using generated selected adversarial samples and analyze the properties of robustness, thus hoping to obtain state-of-the-art DNN models which can successfully defend such attack. However, adversarial examples seem to be bugs in machine learning; such attack is quite irresistible.

Generally, methods of adversarial examples generation can be divided into two types: gradient-based ones and optimization-based ones [20]. For the first type, Szegedy *et. al* [15] first used a gradient-based optimization approach in their work to recognize some DNNs, including state-of-the-art models, are vulnerable to adversarial examples. Thereafter, Goodfellow *et. al* [5] proposed a family of fast methods for generating adversarial examples, also known as *Fast Gradient Sign Method*, which applies a first-order approximation of the loss function to construct adversarial samples [19]. Then, they explained adversarial examples as an existence in a high dimensionality and a result of models being too linear, rather than too nonlinear [5]. In other words, many pixels are required to be changed in the *Fast Gradient Sign Method* [5], which is a kind of limitation. In order to solve this, Su *et. al* [13] proposed a novel method for generating adversarial perturbations, which only requires a single pixel change . Their work proved that DNNs are not only vulnerable to high-dimensional attacks but also to low-dimensional attacks. As for the second type, optimization-based approaches, Nicolas *et al.* [12] introduced the first practical demonstration of an attacker without internal model information and without access to the training data; such attacker is also known as the *black box attack*. They trained a local model to replace the target DNN model, thus keeping the model internals confidential, which is highly suitable for application scenarios in real life.

Additionally, some interesting properties of adversarial examples have also been discovered in previous work. For instance, Gilmer *et al.* [4] researched on the relationship between high-dimensional geometry and adversarial examples, then pointed out that this caused some counter-intuitive phenomena in machine learning: even models with very low test error tend to be vulnerable to slightly chosen perturbations. Afterwards, the empirical and theoretical evidences for adversarial transferability and robustness are reported. Nicolas *et al.* [11] first introduced the transferability of adversarial samples in machine learning, across models both trained by same and different techniques. Alhussein *et al.* [3] first derived fundamental upper bounds on the robustness to perturbations and further led to insights onto the key properties of generative models.

All of the work aforementioned has lead to a further research in the field of adversarial examples; thereby different perspectives on the existence and properties of adversarial examples keep emerging, including the paper [7] we are going to reproduce. Details will be discussed in the following sections.

3 Proposed Approach

3.1 Robust Features Model

The robust features model is loosely based on the work of Tsipras *et al.* [16], which can refer to "robust" and "non-robust" features. In the paper [7] that we reproduce, the authors aim to study the binary classification, where classifier $C : \mathcal{X} \to \{\pm 1\}$ predicts label y given features x.

3.1.1 Feature Definition

Feature space is defined by mapping from the input space $\mathcal{X}$ to the real numbers, thus being $\mathcal{F} = \{f : \mathcal{X} \to \mathbb{R}\}$. In addition, all features are assumed to be zero-mean and unit-variance in order to allow the following definitions scale-invariant.

ρ-useful Features The feature f is ρ-useful ($\rho > 0$) when the feature is correlated with the label in expectation as follows:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[y \cdot f(x)] \geq \rho. \tag{1}$$

$\rho_{\mathcal{D}}(f)$ is then defined as the largest ρ for which feature f is ρ-useful under distribution $\mathcal{D}$. It is notable that a linear classifier trained on ρ-useful features can attain non-trivial generalization performance.

γ-robustly Useful Features The feature f is defined as a robust feature when f still remains γ-useful under certain valid adversarial perturbations Δ. Formally, if we have that

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[\inf_{\delta\in\Delta(x)} y \cdot f(x+\delta)] \geq \gamma. \tag{2}$$

Useful, Non-robust Features The useful, non-robust feature is a feature which is ρ-useful for some ρ bounded away from zero, but is not a γ-robust feature for any $\gamma \geq 0$. While these features may hinder classification accuracy in the adversarial configuration, they can help the accuracy in the standard configuration when the correlation with the label is flipped.

3.1.2 Classification

A classifier C is represented as $C = (F, w, b)$, which consists of features F, a weight vector w and a bias b; a classifier is trained by minimizing a loss function $\mathcal{L}_\theta(x, y)$ on the training set. In their work, the feature set learned by a classifier C is denoted as F_C.

Standard Training Empirical risk minimization (ERM) is applied to minimize the loss function in the standard learning, which aims to decreases the correlation between the weighted combination of the features and the label. It is notable that robust features are same as non-robust features, when minimizing classification loss. Besides, all ρ-useful features can be adopted to decrease the classification loss.

Robust Training Considering that any useful but non-robust features may lead to adversarial vulnerability, such as degrading classification accuracy, it is hard to obtain a robust classifier through ERM in the presence of an adversary. Hence, the authors introduce an adversarial loss function that can discern between robust and non-robust features [9]:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[\max_{\delta\in\Delta(x)} \mathcal{L}_\theta(x+\delta, y)], \tag{3}$$

where Δ are certain adversarial perturbations. As a result, the minimization of this adversarial loss can prevent a classifier from from learning a useful, but non-robust features.

3.2 Finding Robust and Non-Robust Features

In the reproduced paper, the authors prove that both both robust and non-robust features can improve the standard classification, by disentangling these two feature sets. First, they generate a robust dataset that mainly contains robust features, and the robust classifiers are trained based on such robust dataset. The result shows that robustness can be improved after removing certain features, and adversarial vulnerability is caused by non-robust features and is not inherently tied to the standard training framework. In addition, the second dataset is built based on all non-robust features, and can also suffice to train a classifier with good accuracy on the standard testing set. Hence, these non-robust features alone are also sufficient for non-trivial generalizations performance on natural images, which indicates that they are indeed valuable features, rather than artifacts of finite-sample overfitting.

3.2.1 Disentangling Robust and Non-robust Features

Due to the challenges of directly manipulating complex and high-dimensional datasets, the authors determine to develop a robust model and only consider the features that are relevant to such robust model. Given a robust classifier C, a distribution $\hat{\mathcal{D}}_R$ is constructed for which features used by C are equally important as they are on the original distribution $\mathcal{D}$ while ensuring the remaining features are less important. The proposed framework is represented as follows:

$$\mathbb{E}_{(x,y)\sim\hat{\mathcal{D}}_R}[f(x)\cdot y] = \begin{cases} \mathbb{E}_{(x,y)\sim\mathcal{D}}[f(x)\cdot y] & \text{if } f \in F_C, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where F_C is the feature set learned by classifier C.

The training set on the distribution $\hat{\mathcal{D}}_R$ is generated through a one-to-one mapping $x \to x_r$ from the original training set on the distribution $\mathcal{D}$. For example, F_C is the set of activations in the penultimate layer when the classifier is a deep neural network. In order to allow features learned by the classifier are equally important, the feature values of x and x_r should remain similar through the following optimization:

$$\min_{x_r} \|g(x_r) - g(x)\|_2 \tag{5}$$

where x is the original input and g is the mapping from x to the representation layer. The objective is optimized using gradient descent.

Figure 1 shows random samples from the original training set $\mathcal{D}$, the robust training set $\hat{\mathcal{D}}_R$ that contains features learned by a robust classifier, and the non-robust training set $\hat{\mathcal{D}}_{NR}$ that contains features relevant to a standard classifier, respectively. After disentangling features as shown in Figure 2, the authors train the classifiers based on robust and non-robust dataset, respectively. The results indicate that the classifier trained using the robust dataset can report good accuracy in both standard and adversarial settings. In addition, while good standard accuracy can be achieved in the non-robust dataset classification, there are posing challenges to the good robust accuracy.

3.2.2 Non-robust Features for Standard Classification

The previous sections suggest that non-robust features can take an important role in the classification accuracy, when training on the standard dataset. The authors demonstrate that this is not incidental and non-robust features can suffice for standard classification.

They first construct a new training set where the only features that are useful for classification are non-robust features. Each input-label pair (x, y) is modified, and the target class t can be randomly

Figure 1: Random samples from the variants of CIFAR-10 training set [8].

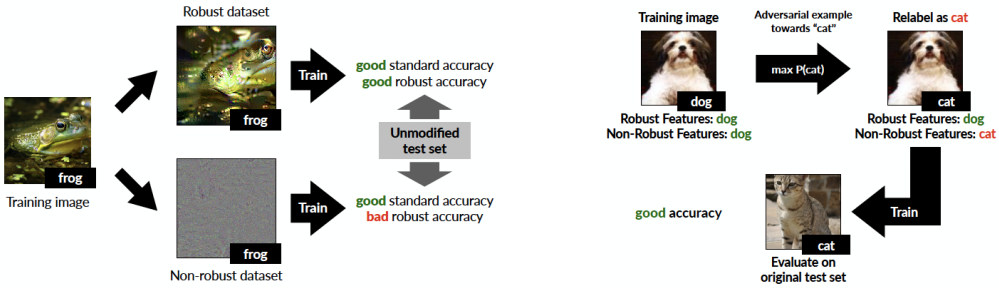

Figure 2: Conceptual diagram of disentangling robust and non-robust features.

or deterministically selected. After that, an adversarial perturbation is introduced to x, which leads to classify x as the target class t by a standard model:

$$x_{adv} = \underset{\|x^{'}-x\| \leq \varepsilon}{\arg\min}\ L_C(x^{'}, t), \tag{6}$$

where L_C is the loss function and ε is a small constant. The pairs (x_{adv}, t) are the new training set. It is notable that the robust features of x_{adv} are still correlated with original class y rather than t and human still recognize the original class, since $\|x^{'} - x\|$ is small.

In the case where target class t is randomly selected, the robust features are originally uncorrelated with t, and will be slightly correlated with the class after introducing the adversarial perturbation. A new dataset $\hat{\mathcal{D}}_{rand}$ is generated as follows:

$$\mathbb{E}_{(x,y)\sim\hat{\mathcal{D}}_{rand}}[y \cdot f(x)] \begin{cases} > 0 & \text{if } f \text{ non-robustly useful under } \mathcal{D}, \\ \simeq 0 & \text{otherwise.} \end{cases} \tag{7}$$

In the second case where target class t is deterministically selected based on original class y, the robust features in the original training set can take an important role on the new dataset and also pose challenges to the generalization on the standard testing set. A new dataset $\hat{\mathcal{D}}_{det}$ is constructed such that

$$\mathbb{E}_{(x,y)\sim\hat{\mathcal{D}}_{det}}[y \cdot f(x)] \begin{cases} > 0 & \text{if } f \text{ non-robustly useful under } \mathcal{D}, \\ < 0 & \text{if } f \text{ robustly useful under } \mathcal{D}, \\ \in \mathbb{R} & \text{otherwise } (f \text{ not useful under } \mathcal{D}). \end{cases} \tag{8}$$

3.2.3 Transferability From Non-robust Features

An interesting finding of adversarial examples is that they can transfer across models with different architectures and independently sampled training sets [14, 10, 1]. Considering that adversarial examples can arise through perturbing well-generalizing, yet brittle features, various classifiers trained on independent samples from data distribution are likely to utilize similar non-robust features. As a result, adversarial examples constructed by exploiting the non-robust features learned by one classifier will transfer to other classifiers utilizing similar features. This finding also suggests that architectures which has better classification accuracy on the standard testing set are more likely to learn similar non-robust features to the original classifier.

4 Dataset Description and Reproduced Tasks

4.1 Dataset Description

In our experiments, all models are evaluated on the whole CIFAR-10 dataset (Canadian Institute For Advanced Research) [8]. The CIFAR-10 dataset is a collection of images that are most widely used to train machine learning and computer vision models [17]. The dataset consists of $60,000$ RGB images of shape 32×32 in 10 different classes, which are airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck, respectively; each class includes $6,000$ images. Several recent papers achieved a start-of-the-art performance on CIFAR-10 dataset; for instance, Martin *et al.* [18] achieved the classification accuracy of 98.67%, while Yanping *et al.* [6] reported around 99.00% accuracy in their classification on CIFAR-10 dataset.

4.2 Reproduced Tasks

In the reproduced tasks, we first constructed a new robust dataset for a ResNet-50 through generating untargeted adversarial examples on the original CIFAR-10 dataset. The generation approach has been detailedly discussed in Section 3.2.2, and adversarial examples could be generated by introducing an adversarial perturbation ϵ into the original dataset x. After randomly generating adversarial examples, the original robust features would be slightly correlated with the target class.

Thereafter, based on the new robust dataset we generated in the first step, we designed, implemented and evaluated the ResNet-50 classifier baseline of the reproduced paper [7]. Specifically, several robust datasets could be generated based on the different values of adversarial perturbation ϵ, and each dataset is performed to train a ResNet-50 classifier. Besides, we also designed, implemented and evaluated the VGG-16, InceptionV3 and DenseNet-121 classifiers as the extension of reproduced model, in order to find more robust classifier. The accuracy of these classifiers will be shown in Section 5.

5 Experiments Setup and Simulation Results

The illustration for the untargeted adversarial examples is shown in Figure 3. The first row displays part of the raw images in CIFAR-10 dataset, while the second row shows images after adversarial perturbation implementation. It is obvious to see that, some labels are changed in the second row even though the entities in the images can still be recognized by humans. This is the first step in our experiments, which constructs a robust dataset for our successive studies.

In addition to the untargeted adversarial examples generation, we finished our experiments on four different aspects to study the influences of hyperparameters; thus obtaining a comprehensive understanding of the approaches and ideas in the paper [7]. The settings of the rest hyperparameters are same as the configurations of the reproduced model [7].

Experiments on normalization During the robust dataset construction procedure, the paper [7] normalized the gradient at each step of PGD (projected gradient decent) in an $L2$-norm. We first

Figure 3: Illustration for untargeted adversarial examples.

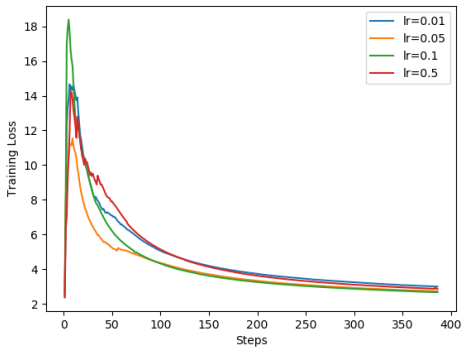

Figure 4: Convergence lines for different learning rate settings.

reproduced their work, and then removed their normalization, instead, implementing an $Linf$-norm for each step, for a comparison. As the results displayed in Table 4, generally speaking, models with $Linf$-norm reach a higher accuracy than models with $L2$-norm in the same hyperparameters setting.

Experiments on ϵ Thereafter, we changed the ϵ value (a threshold in robust dataset construction) of the model and compared it with the original one mentioned in the paper [7]. Both training set and testing set are related to an adversarial perturbation; thereby, there are two ϵ we need to consider in our experiments: ϵ_{train} and ϵ_{test}. For CIFAR-10 with $L2$ normalization, we set both ϵ_{train} and ϵ_{test} in 0, 0.25, 0.5, and 1.0 respectively. While for that with $Linf$ normalization, we set ϵ_{train} in $\frac{0}{255}$ and $\frac{8}{255}$, with ϵ_{test} in $\frac{0}{255}$, $\frac{8}{255}$, and $\frac{16}{255}$. It is notable that the robust dataset becomes a non-robust dataset when $\epsilon_{train} = 0$, since no adversarial perturbation is added. The results are shown in Table 1 and 2. It is obvious to notice that robust models ($\epsilon_{train} \neq 0$) show a much better performance than standard models ($\epsilon_{train} = 0$), no matter what kind of normalization being implemented.

Experiments on learning rate Moreover, we modified the learning rate on the original model and compared the classification accuracy on four different settings: 0.01, 0.05, 0.1, and 0.5. As the results shown in Table 3, models achieve a great performance when setting learning rate as 0.5, which can reach 99.13%. Additionally, we also studied and analyzed the influence of different learning rates on the convergence of the model. Based on the convergence lines shown in Figure 4, we find models can converge more quickly when learning rate equals to 0.1. Therefore, to sum up the two findings aforementioned, we recommend to set learning rate as 0.1 or 0.5.

Experiments on classifier Finally, as an attempt to extend the reproduced model [7] and to further find potential robust classifier, we designed, implemented and evaluated different classifiers, i.e. VGG-16, InceptionV3, and DenseNet-121 classifier. As shown in Table 4, the results indicate that the DenseNet-121 classifier performs the best overall, reaching a surprising accuracy of 90.49%.

Table 1: CIFAR-10 L2-norm Accuracy (%)

$\epsilon_{test} \setminus \epsilon_{train}$	0	0.25	0.5	1.0
0	75.84	**99.11**	96.70	98.73
0.25	74.68	98.89	97.90	97.83
0.5	71.50	96.77	96.50	96.27
1.0	49.76	92.53	92.52	92.57

Table 2: CIFAR-10 Linf-norm Accuracy (%)

$\epsilon_{test} \setminus \epsilon_{train}$	0/255	8/255
0/255	80.76	93.75
8/255	53.39	98.35
16/255	48.45	**98.63**

Table 3: CIFAR-10 Robust Accuracy (%) Comparison For Different Learning Rates

CIFAR-10 L2-norm Accuracy				
Model \ Learning rate	0.01	0.05	0.1	0.5
Standard training ($\epsilon_{train} = 0$ and $\epsilon_{test} = 0$)	71.41	75.42	71.36	81.73
Robust training ($\epsilon_{train} = 0.5$ and $\epsilon_{test} = 0$)	97.96	97.51	98.26	**99.13**
Robust training ($\epsilon_{train} = 0.5$ and $\epsilon_{test} = 0.5$)	96.60	97.66	97.54	98.54
CIFAR-10 Linf-norm Accuracy				
Standard training ($\epsilon_{train} = 0$ and $\epsilon_{test} = 0$)	75.16	78.97	80.50	83.87
Robust training ($\epsilon_{train} = 8/255$ and $\epsilon_{test} = 0$)	97.85	97.28	98.37	98.73
Robust training ($\epsilon_{train} = 8/255$ and $\epsilon_{test} = 8/255$)	97.26	98.61	98.62	96.42

Table 4: CIFAR-10 Robust Accuracy (%) Comparison For Different Models

Model	Accuracy
ResNet-50	75.84
VGG-16	74.58
InceptionV3	81.49
DenseNet-121	**90.49**

6 Discussion and Conclusion

In this project, we first summarized the paper [7] which proposed that adversarial examples can be directly attributed to the presence of non-robust features. Specifically, the proposed robust features model categorized features in the ρ-useful,γ-robustly useful and useful, Non-robust Features. After that, the proposed model disentangled robust and non-robust features, and generated the new dataset where the only features that are useful for classification are non-robust features, through introducing adversarial perturbation to the original features. The accuracy of classifier trained on such new dataset can support that non-robust features alone are sufficient for good classification and generalization.

After summarizing their works, we constructed a new dataset for a ResNet-50 through generating untargeted adversarial examples on the whole CIFAR-10 dataset. This new dataset was applied to design, implement and evaluat the ResNet-50 classifier baseline of the reproduced paper [7]. Due to the challenges of computational resources, our ResNet-50 classifier could only be trained within the limited iterations and reported 75.84% accuracy in the standard classification. In addition, we also modified the classifier hyperparameters to explore how these hyperparameters impact the accuracy, such as normalization approach, value of adversarial perturbation and learning rate. We found out that $Linf$ is a better setting than $L2$ for normalization approach, and models trained on robust dataset perform much better than those trained on non-robust dataset. Moreover, a learning rate of 0.5 tends to lead a better accuracy, while a learning rate of 0.1 helps a more fast convergence. Furthermore, all VGG-16, InceptionV3 and DenseNet-121 classifiers were designed, implemented and evaluated on the whole CIFAR-10 dataset, as the extension of reproduced ResNet-50 classifier [7] , in order to find potential better classifier. The results indicated that DenseNet-121 classifier reported the best accuracy of 90.49% in the standard classification. Hence, DenseNet-121 can be suggested as one future direction to improve the classification accuracy.

References

[1] Zachary Charles, Harrison Rosenberg, and Dimitris Papailiopoulos. A geometric perspective on the transferability of adversarial directions. *arXiv preprint arXiv:1811.03531*, 2018.

[2] Christoph Molnar. Interpretable machine learning: A guide for making black box models explainable., 2019. [Online; accessed 12-December-2019].

[3] Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. Adversarial vulnerability for any classifier. In *NeurIPS*, 2018.

[4] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian J. Goodfellow. Adversarial spheres. *ArXiv*, abs/1801.02774, 2018.

[5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014.

[6] Yanping Huang, Yonglong Cheng, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *ArXiv*, abs/1811.06965, 2018.

[7] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *ArXiv*, abs/1905.02175, 2019.

[8] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[10] 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*, 2016.

[11] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *ArXiv*, abs/1605.07277, 2016.

[12] Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ASIA CCS '17*, 2016.

[13] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23:828–841, 2017.

[14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[15] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

[16] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[17] Wikipedia contributors. Cifar-10 — Wikipedia, the free encyclopedia, 2019. [Online; accessed 14-December-2019].

[18] Martin Wistuba, Ambrish Rawat, and Tejaswini Pedapati. A survey on neural architecture search. *ArXiv*, abs/1905.01392, 2019.

[19] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Xiaodong Song. Generating adversarial examples with adversarial networks. In *IJCAI*, 2018.

[20] P. Yu, K. Song, and J. Lu. Generating adversarial examples with conditional generative adversarial net. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 676–681, Aug 2018.

[21] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30:2805–2824, 2017.