

---

# Brain-inspired Reverse Adversarial Examples

---

Shaokai Ye<sup>1</sup> Sia Huat Tan<sup>1</sup> Kaidi Xu<sup>2</sup> Yanzhi Wang<sup>2</sup> Chenglong Bao<sup>1</sup> Kaisheng Ma<sup>1</sup>  
<sup>1</sup>Tsinghua University, China  
<sup>2</sup>Northeastern University, USA

## Abstract

A human does not have to see all elephants to recognize an animal as an elephant. On contrast, current state-of-the-art deep learning approaches heavily depend on the variety of training samples and the capacity of the network. In practice, the size of network is always limited and it is impossible to access all the data samples. Under this circumstance, deep learning models are extremely fragile to human-imperceivable adversarial examples, which impose threats to all safety critical systems. Inspired by the association and attention mechanisms of the human brain, we propose reverse adversarial examples method that can greatly improve models' robustness on unseen data. Experiments show that our reverse adversarial method can improve accuracy on average 19.02% on ResNet18, MobileNet, and VGG16 on unseen data transformation. Besides, the proposed method is also applicable to compressed models and shows potential to compensate the robustness drop brought by model quantization - an absolute 30.78% accuracy improvement.

## 1 Introduction

During the process of the neural networks application systems being deployed, one problem that keeps standing out is systems' robustness and safety. Restricted by robustness, many systems encounter difficulties during deployment or utilization. For instance, autonomous vehicles - a fatal accident occurred on a Tesla vehicle that is under autopilot mode, and the statement from Tesla says that "neither autopilot nor the driver noticed the white side of the tractor trailer against a brightly lit sky, so the brake was not applied" [25]. In this case, the light condition can be seen as misleading perturbation to inputs that lead to the wrong decision in autopilot system. V2X (Vehicles to X) - for new camera devices added on the infrastructure, the model needs to be retrained from scratch to adjust to the light condition; Smart shopping - a camera placed with light condition A needs to be retrained based on a model trained from data in light condition B. Theoretically, if catastrophic forgetting can be overcome, robustness can be solved with enormous amount of data and a network of an infinite volume. While in reality, most neural networks are deployed in scenarios with limited resources. In order to better adapt to the limited computing resources, many techniques are even proposed to further reduce the volume - storage and computation - of the network, for instance, pruning [10, 26], quantization [9, 30] and etc., which will further reduce the robustness of neural networks [8, 16].

Another critical issue for neural networks is the robustness for adversarial examples[24]. By adding maliciously crafted adversarial perturbation to inputs, attackers can make deep learning models mis-classify, even if the perturbation is imperceivable to human [4, 7, 15].

In general, robustness and adversarial robustness are used to evaluate the capability of deep neural networks to accurately classify examples that are affected either by data transformation or maliciously crafted adversarial perturbation.

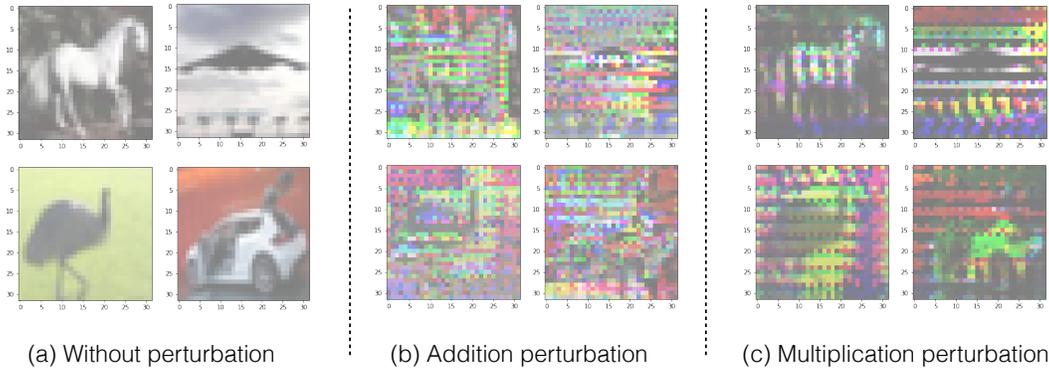


Figure 1: Visualization of two types of perturbations. (a) are original images. (b) are addition reverse adversarial perturbations. (3) are multiplication reverse adversarial perturbations. Compared to addition reverse adversarial perturbations, multiplication reverse adversarial perturbations carry more semantic meanings.

## 1.1 Motivations and Contributions

### 1.1.1 Brain-inspired robustness

As to the aforementioned issues in neural networks, human brain is more robust. From the perspective of psychology, there is a term called associationism [19] which refers to the association formed between the concepts of things. It can be classified as simple concept and complex concept. Two features of the associationism are contiguity which means that things or events with spatial or temporal proximity tend to be associated in the mind, and similarity which demonstrates that thought of one event tends to trigger the thought of a similar event. These characters are, to some extent, identical to what we have in system. Inspired by associationism, we propose a calibrator that can perform associationism-similar functions in neural networks.

In biology, all animals make decisions with extremely limited resources which are the energy extracted from food. In order to save energy for computation, the brain actually evolves a mechanism called attention [5, 29, 6]. When a picture is reflected by eyes, only a small part of it is processed by the conscious processing system, while the rest are slightly computed and memorized by the sub-conscious system and some control-related-tasks are taken over by the muscle memory. In this way, only the most important parts of the data are processed with extra attention.

Inspired by attention mechanism of human brain, we propose a multiplication based operation to merge benign perturbation generated by the calibrator to inputs. Figure 1 shows the comparison between traditional addition method and attention-inspired multiplication method. Note that these are pure perturbations, not perturbations merged into original inputs. Interestingly, the multiplication based perturbation carries more semantic meanings compared to addition based perturbation. This shows that the proposed way of merging perturbation creates perturbation that focuses more on modifying label-correlated area of the inputs.

### 1.1.2 Existence of adversarial examples

Deep learning models are so powerful in extracting features from inputs, so that they are ultra sensitive to inputs. Any alteration caused by either scenes or perturbations generated by adversarial attacks can result in severe impacts on the networks. To overcome the problems caused by sensitivity of the networks, the networks are trained against data with more complexity [7, 18]. We propose using a calibrator network that can capture the connection between sets of data to improve the robustness. If we view adversarial attacks as attacker PUSHING AWAY some input data from the true class to a target class (wrong for tasks human care), then it's likely that we can PULL input data TOWARDS the true class. In other words, we wish to optimize inputs such that they can be easily recognized by the given classifier.

As an extreme case study, a LeNet model is initialized with completely random weights, then tested on MNIST and CIFAR10 data set. The model is left untrained and we perform reverse adversarial attack by modifying previous adversarial attack algorithm [15] to craft reverse adversarial examples. Table 1 shows that we are able to achieve 87% and 53% accuracy without updating the randomized models. This example demonstrates that the inputs can be optimized towards given neural networks.

Table 1: **Results of randomly initialized weights under reverse adversarial attacks** (Acc. in %)

Model	Accuracy with an untrained randomly initialized network	Accuracy of untrained models under reverse adversarial attack
LeNet on MNIST	10.57	87.27
LeNet on CIFAR10	10.31	53.38

**Contributions:** Our contributions are summarized as follows:

1. To the best of our knowledge, this is the first attempt to discuss and utilize adversarial examples reversely. Inspired by associationism, the proposed framework includes a generative network called calibrator to generate benign perturbation to pull the unlearned features from unseen data to the learned ones. Furthermore, the multiplication based perturbation is imposed to mimic the attention mechanism in the human brain,
2. The results show that the proposed method can improve models' performance on unseen data with strong difference (data B2 in Table 2) by 19.30%, 18.11%, 19.66% for MobileNet, ResNet18, and VGG16 respectively. For compressed models (2 bits), models' performance can increase by 29.96%, 28.76%, and 20.67% on unseen data.
3. The proposed reverse adversarial examples method has broad potential applications. Without retraining the main model, it can enhance scenario-specific robustness when the main model is deployed in ASIC for energy efficiency, and the calibrator is deployed in ARM Core for scenario-specific flexibility.

## 2 Related work

**Adversarial attacks.** Adversarial examples [24, 3] are created by adding human-imperceptible perturbation to inputs. These malicious examples are transferable [20, 21, 17, 28], and are even effective in physical world [2, 23].

There are many methods for generating adversarial examples, such as FGSM [7], IFGSM [15] and C&W [4]. However, adversarial examples have values beyond malicious attacks. Inspired by the vulnerable nature of neural networks, the calibrator we proposed is able to perform reverse adversarial attacks so that powerful attack methods can turn to benefit neural networks.

**Adversarial training.** Goodfellow *et al.* [7] train a mixture of nature examples and adversarial examples to improve models' robustness. Madry *et al.* [18] formulate adversarial training as a min-max optimization problem and solve it iteratively, which is the only method that passes the obfuscated gradient check [1].

Those methods either partly or fully use adversarial examples as training data, resulting in models that are robust to adversarial examples, but along with cost of accuracy degradation in nature examples. In resource constrained systems, requiring a small model to train adversarial examples (or a mixture of nature and adversarial examples) is not feasible.

**Generative networks for adversarial examples.** Prior works [27, 13] use generative models to produce adversarial examples. Those work are able to generate effective adversarial examples in feed-forward process. However, those methods only focus on generating malicious perturbation for attacks, rather than generating reverse adversarial examples for benign uses.

## 3 Proposed brain-inspired calibrator

One common deployment process for deep learning models usually starts with training a large model in one specific training dataset, which is assumed to be close to the applications. Once the model

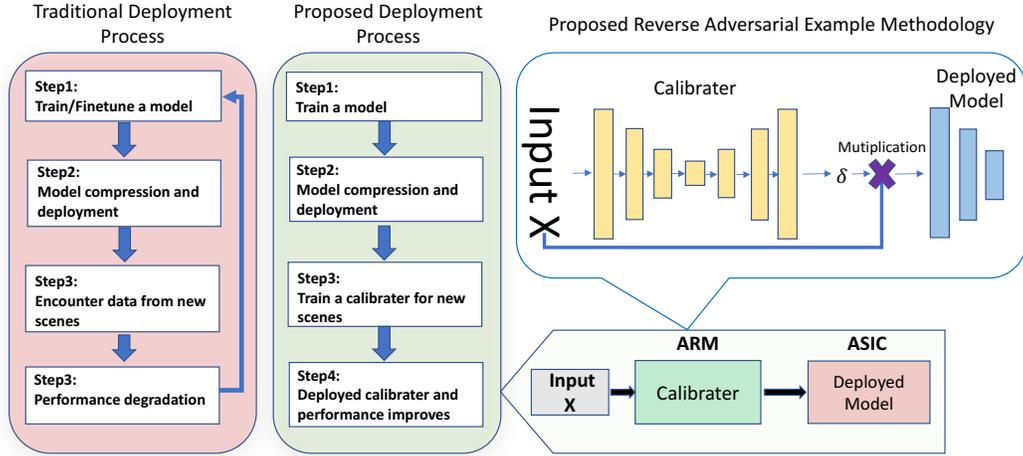


Figure 2: Overview of proposed reverse adversarial example structure

is trained, it is deployed in hardware followed by the model compression on the same training set. However, the real world is complicated - the assumption that the training data and data in deployment are independently and identically distributed can seldomly hold: scenes might change. For example, in self-driving training, the data collected in a city that is always sunny is very different from those of a city that rains a lot. In such cases, the performance degradation occurs. In order to compensate the performance degradation, the traditional approach is either adding the new data to fine-tune the original model or training a completely new model that is only good for the new scene (see the left part of figure 2). In both cases, the training cost of the main model is huge and this issue scales up quickly when the size of the main model and the number of deployed devices are increased.

The proposed framework approaches to this problem in a very different way: instead of asking models to fit every possible scene, it's more feasible to train a calibrator with new data and help the main model understand data from new scenes. The middle part of Figure 2 demonstrates our proposed deployment process. Instead of training the main model again, we propose to train a light weight generative calibrator when the additional unseen data have been applied. Compared to the traditional deployment process, our approach significantly accelerates the deployment process. Meanwhile, the proposed generative calibrator is feasible to run in an ARM embedded processor due to its simplicity. Once the deployed device is switched for different scenarios, it is very flexible to update the calibrator running on ARM, rather than the main model running on ASIC. In the following context, we introduce the details in training our proposed calibrator.

### 3.1 Brain-inspired calibrator

In this section, we define  $f(\cdot, \theta)$  to be a neural network with parameters  $\theta$  and  $c(x) = \arg \max(f(x))$  to be the output classification result of the neural network. Given a training set  $\mathcal{X}_1 = \{(x_i, y_i)\}_{i=1}^n$ , the network parameter  $\theta$  is obtained via solving the minimization:

$$\theta^* \in \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n \text{loss}(f(x_i, \theta), y_i), \quad (1)$$

where loss is the loss function defined by users. For a given network  $f(x, \theta^*)$ , typical adversarial attack methods find the perturbation  $\delta$  via the optimization model [7, 15, 4]:

$$\min_{\delta} \|\delta\|, \text{ s.t. } c(f(x + \delta, \theta^*)) \neq c(f(x, \theta^*)) \text{ or } \max_{\delta} \ell(f(x + \delta, \theta^*), f(x, \theta^*)), \text{ s.t. } \|\delta\| \leq \Delta, \quad (2)$$

where  $\ell$  measures the difference between the network prediction of  $x + \delta$  and  $x$  and  $\Delta$  is the perturbation range. It is noted that the adversarial attacks always exists for a given classifier, i.e. there exists bad samples in the local region of the given data. Motivated by this observation, we propose to reverse the process of generating adversarial attacks such that to find a better perturbation of the given sample which can be easily recognized by the trained classifier. More concretely, given a set of unseen data  $\mathcal{X}_2 = \{(x'_i, y'_i)\}_{i=1}^m$  where  $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$ , denote the benign perturbation by  $G(x, \theta_G)$

where  $\theta_G$  are the parameters of  $G$ , we aim at finding the perturbation  $\delta = G(x, \theta_G)$  via minimizing the loss

$$\min_{\theta_G} L_G = \frac{1}{m} \sum_{i=1}^m \text{loss}(f(x'_i \odot G(x'_i, \theta_G), \theta^*), y'_i), \quad (3)$$

where  $\odot$  is the point-wise multiplication. Compared to the adversarial attack (2), our calibrator defers from two perspectives: (i) we perform reverse adversarial attack by requiring the generative model to output  $\delta$  to minimize classification loss of the main model  $f(x \odot \delta)$  in the new dataset  $\mathcal{X}_2$ ; (ii) instead of the addition of the perturbation in (2), point-wise multiplication mimics the attention mechanism in brain as shown in prior work [5, 29, 6].

---

**Algorithm 1** Training for proposed calibrator  $G$

---

- 1: **Inputs:** Dataset  $\mathcal{X}_1$  and  $\mathcal{X}_2$ , the main classifier  $f(\cdot, \theta)$  and calibrator  $G(\cdot, \theta_G)$
  - 2: **Outputs:**  $\theta^*$  and  $\theta_G^*$
  - 3: Train the main classifier  $f(\cdot, \theta)$  in dataset  $\mathcal{X}_1$  via solving (1).
  - 4: // Train the calibrator  $G(\cdot, \theta_G)$  in dataset  $\mathcal{X}_2$  via solving (3)
  - 5: **for** batch in Dataloader(dataset  $\mathcal{X}_2$ ) **do**
  - 6:      $data, target = \text{getData}(\text{batch})$  // get data and true labels in a batch
  - 7:      $\delta = G.\text{getOutput}(data)$  // get output from the calibrator
  - 8:      $F = f(\cdot, \theta).\text{getSoftMaxOutputs}(data \cdot \delta)$  // get softmax output from the classifier
  - 9:      $index = \text{argmax}(target)$  // get the index of where the true label has maximal value 1
  - 10:      $L_G = 1 - F[index]$  // add loss if the softmax output of classifier on that index is away from 1
  - 11:      $grad = \text{Opt.getGradient}(L_G)$  // calculate the gradients for  $G$  based on the loss
  - 12:      $\text{weightUpdate}(G, grad)$  // update network  $G$  based on the gradients
- 

## 4 Results

### 4.1 Experiment setup

In this section, we evaluate the proposed benign perturbation calibrator based on CIFAR10 data set, using MobileNet [12], VGG16 [22], and ResNet18 [11] models (also referred as main models). Our implementation is based on PyTorch and we use 4 different data augmentation functions in PyTorch to define the so called A/B scenario setting. We first assume that training data and test data for CIFAR10 are identically and independently distributed. We refer data that are transformed via random crop resize and random horizontal flip as scenario A, and refer data that are transformed via random rotation and random color jittering as scenario B. For further comparison, we also derive B1 and B2 as scenario B under increasing strength of data transformations. For scenario B1, maximum rotation is 15, maximum change of brightness is 0.8, maximum change of contrast is 0.8, maximum change of saturation is 0.8. And for scenario B2, the maximum rotation is 20, maximum change of brightness is 2, maximum change of contrast is 2 and maximum change of saturation is 2. For scenario A, we only use PyTorch’s default settings. All main models are trained under scenario A in training data, and tested under scenario B1/B2 in test data. The calibrators are trained under scenario B1/B2 in training data and used to support main models under scenario B1/B2 in test data. The backbone of the generative model is consisted of 3 down-sampling convolutional layers, followed by a number of residual blocks and 3 up-sampling convolutional layers. The sizes of the calibrators are chosen to be one tenth of the main models. To control the size of the calibrator, we vary the number of channels within residual blocks and number of residual blocks. For example, to have a calibrator that has one tenth parameter the size of MobileNet, we use 2 residual blocks with 18 channels. The solver chosen for training main models and calibrators is ADAM[14]. Training for calibrators takes 200 epochs and the initial learning rate is set to be 0.0002 before 50 epochs, 0.0001 at 50-100 epochs, 0.00005 at 100-150 epochs, and 0.00002 at 150-200 epochs. During training of calibrators, main models are only used for feed-forward.

### 4.2 Robustness improvement brought by reverse adversarial examples method

**Results on uncompressed models:** Table 2 shows the robustness increase brought by the proposed method. In Table 2 Column 3 lists models’ test accuracy on clean test data without any data augmentation. When models are tested under scenario B1 and B2 in test data, models’ accuracy drop

Table 2: **Robustness improved by proposed method** (Accuracy in %) on Cifar10

Test Domain	Networks	Ideal Case Accuracy: Training on A+ Test on A	Baseline Accuracy: Training on A No Calibrator Test on B1/B2	Accuracy: Training on A Calibrator-mul. on B1/B2 Test on B1/B2	Model Size (MB)	Calibrator Size (MB)
Test on B1*	MobileNet	90.88	65.78	<b>75.59(+9.81)</b>	12.88	1.24
	ResNet18	93.96	73.84	<b>78.62(+4.78)</b>	44.80	4.32
	VGG16	91.61	70.11	<b>80.82(+10.71)</b>	58.80	5.48
Test on B2#	MobileNet	90.88	43.78	<b>63.08(+19.30)</b>	12.88	1.24
	ResNet18	93.96	48.35	<b>66.46(+18.11)</b>	44.80	4.32
	VGG16	91.61	48.38	<b>68.04(+19.66)</b>	58.80	5.48

+ A: Random crop resize and random horizontal flip.

\* B1: Maximum rotation is 15 degree, maximum change of brightness, contrast, saturation are 0.8.

#B2: Maximum rotation is 20 degree, maximum change of brightness, contrast, saturation are 2.

Table 3: **Robustness improvement on compressed models** (Accuracy in %) on Cifar10

Number of bits	Networks	Quantized models Acc. on testset without data augmentation	Baselines: Quantized Models Acc. on B2	Quantized Model Acc. on B2 with Calibrator	Model Size (MB)	Calibrator Size (MB)
3 bits	MobileNet	83.69	35.63	<b>66.33(+30.70)</b>	3.22	0.31
	ResNet18	90.00	39.41	<b>70.18(+30.77)</b>	11.20	1.08
	VGG16	91.53	46.69	<b>67.46(+20.77)</b>	14.70	1.37
2 bits	MobileNet	77.96	34.33	<b>64.29(+29.96)</b>	1.21	0.31
	ResNet18	89.87	41.4	<b>70.16(+28.76)</b>	4.20	1.08
	VGG16	89.11	45.44	<b>66.11(+20.67)</b>	5.51	1.37

about 20% to 40%, as shown in Column 4 in Table 2). We consider random rotation and random color jittering as common environmental factors in real world and this result demonstrates that deep learning models lack of robustness in our settings. Column 5 shows that by using the proposed calibrator, we can achieve 4.8% to 19.66% accuracy boost (Column 5 in Table 2), with only little overhead brought by the calibrator (Column 6&7 in Table 2).

### 4.3 Robustness improvement on compressed models

**Results on compressed models:** In this subsection, we demonstrate the effectiveness of the proposed calibrator on compressed models. Main models’ weights are quantized to 2/3 bits using deep compression[9]. To ensure that overhead brought by our calibrators is still low compared to the main models, we quantize them to 8 bits without any quantization training. Note that the calibrators’ training can naturally leverage the accelerated inference speed of main models. In comparison between Column 4 in Table 2 and Column 3 in Table 3, we observe that quantized models suffer significantly more accuracy drop under scenario B2 in test data. We consider this phenomenon as an indicator that model compression hurts models’ robustness in our settings. Table 3 Column 5 shows that when deployed the calibrators, the models’ performance on B2 improves significantly.

Table 4: **Transferability test** (Accuracy in %) on Cifar10 on B2

Tested on \ Calibrator trained on B2	MobileNet	ResNet18	VGG16
MobileNet(43.78)	\	56.5(+12.72)	51.61(+7.83)
ResNet18(48.35)	64.13(+15.78)	\	61.42(+13.07)
VGG16(48.38)	58.27(+9.89)	57.58(+9.19)	\

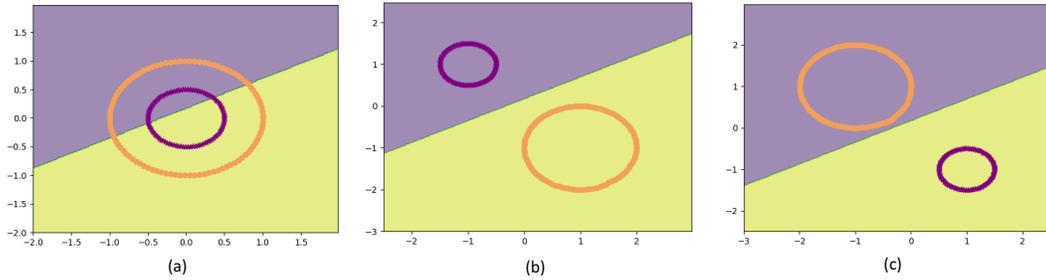


Figure 3: Visualization that shows adversarial attacks and reverse adversarial attacks are symmetric. (a), Results of a linear classifier - a linear classifier’s decision boundary on non-linear-separable data. (b), Results by reverse adversarial examples - the benign perturbation moves inputs to correct regions of the linear classifier’s decision boundary. (c), Results by adversarial examples - data are moved to wrong regions, resulting in 100% misclassification, similar to adversarial attack.

## 5 Discussion

### 5.1 The symmetry between adversarial attacks and reverse adversarial attacks

Using 2-dimensional synthetic data as an example, we visualize the proposed method and demonstrate that adversarial examples and reverse adversarial examples are essentially symmetric. Note that in this subsection, we use addition based perturbation to help demonstrate the symmetry phenomenon. In Figure 3(a), it can be observed that the synthetic data form two concentric circles that represent two classes of data that are not linear separable. A linear classifier is applied to do classification on data, and the plotted decision boundary shows that the linear classifier is not able to fully separate data. In Figure 3(b), we train a calibrator on these data and generate benign perturbation to help the linear classifier do classification. It is clearly observed that after perturbation are applied on input data, data are moved to different regions and become linear separable. In this example, we show that the calibrator is able to optimize inputs towards the linear classifier’s decision boundary, solving a problem that is impossible for a linear classifier in an elegant way. In Figure 3(c), the calibrator has been used reversely. It is noted that data are moved to opposite locations compared to Figure 3(b), demonstrating that the reverse use of our calibrator are able to perform adversarial attacks. This example shows that adversarial attacks and reverse adversarial attacks might be symmetric, as attacks harm the performance of models and reverse attacks benefit models.

### 5.2 Proposed calibrator improves models’ robustness against unseen data

In Figure 4(a), we create synthetic data that form two overlapping ellipses (data are those orange dots and purple dots instead of the regions). Figure 4(c) helps to understand our setting: data from both ellipses form what can be seen as overlapping region. In this subsection, data that are on the non-overlapping region are chosen as training data for the main model, and data on overlapping regions are chosen as training data for the calibrator. Here, a MLP(Multiple Layer Perceptron) is used as the main model to classify its training data. Not surprisingly, it achieves 100% accuracy as shown in Figure 4(a). However, if we ask MLP to classify data that were hidden from its training, the resulted accuracy is as low as 0%. This example represents what commonly happens in real world scenarios. In practice, as shown in Figure 4(b), once the unseen data are realized, the model is put to finetune against those added data. Even in this costly way, the model can only achieve 18.75% accuracy in the added data. On contrast, Figure 4(d) shows that if the calibrator is applied, without needing to finetune the main model, the calibrator is able to transfer the unseen data so as to fit the trained main model. As a result, the main model achieves 68.7% accuracy in data that was unseen by the main model. This example shows the true power of the proposed method: the deployment of calibrator greatly improves models’ robustness against unseen data and reduces the deployment cost. A deep learning model deployed with the calibrator does not have to repeat the costly recycle process in the present of data from new scenes, which makes it possible to deploy neural networks in a completely new fashion.

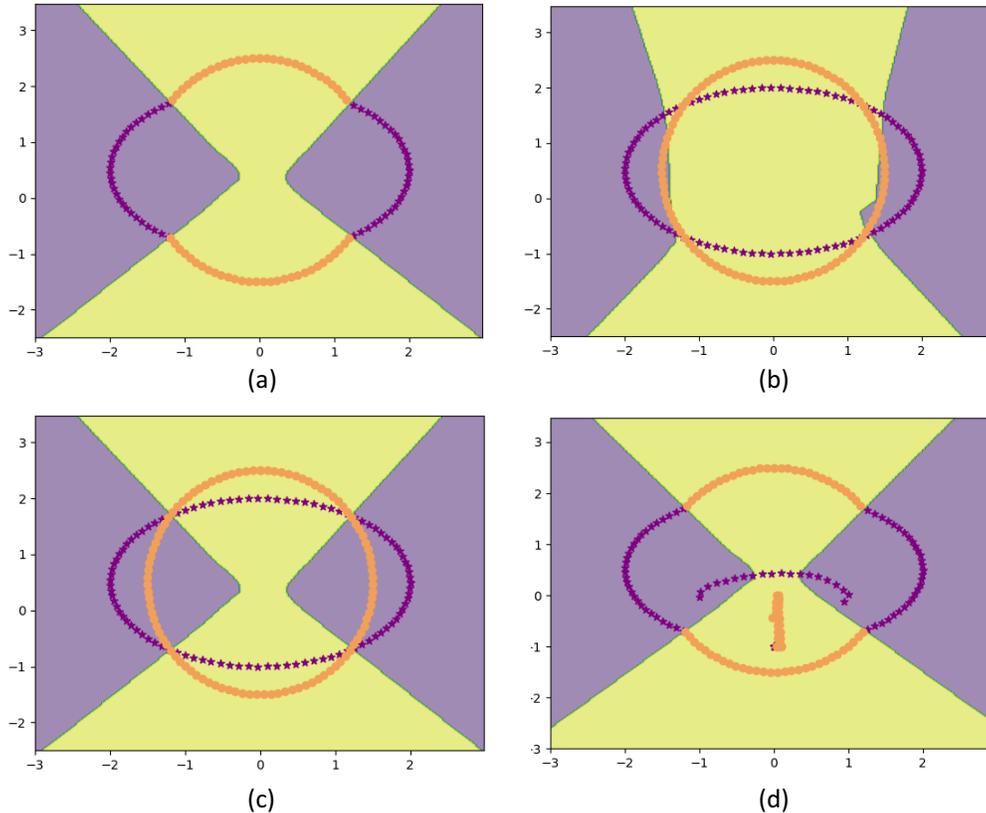


Figure 4: Visualization of how the calibrator handles unseen data. (a), MLP achieves 100% classification on partial data, depicted by dots that are inside decision boundary (Note the difference between (a) and (c)). (b), Results by a MLP trained with full data. The accuracy is 63% for the whole data. (c), Based on the main model in (a), data hidden in (a) are used to train a calibrator. (d), Results by the main model trained in (a) and the calibrator. Test accuracy on excluded data by the main model is 68.7%. For comparison, for unseen data, (a) accuracy is 0%, and (b) accuracy is 18.75%. Benign perturbation move the previously excluded data towards to MLP’s decision boundary.

### 5.3 Reverse adversarial examples are transferable

So far, all experiments are performed in a way that every calibrator is trained with a main model it targets for. It’s natural to raise a question whether it’s necessary to train a calibrator for every main model individually. In Table 4, we show that just like adversarial examples that are transferable [20, 21, 17], reverse adversarial examples are transferable as well. Columns in Table 4 list main models used for training the calibrators and the rows list the main models used for testing their adaptivity with calibrators trained with models from quite different architectures. Here, we keep the A/B setting and test models’ performance under scenario B2 in the test set of CIFAR10. The results show that all models’ performance improve significantly with the use of the calibrator, regardless of which models the calibrators are trained with.

## 6 Conclusion

In this work, we show that it is possible to create reverse adversarial examples and learn a generative model called calibrator to construct them. Unlike adversarial examples that are mainly used for attacking deep learning models, reverse adversarial examples can significantly improve models’ robustness in unseen data transformations, which assembles human brain that associates new knowledge with old knowledge. Furthermore, we borrow the idea of attention mechanism in human brain and introduce attention-inspired multiplication perturbation, which carries more semantic meaning. The proposed method takes a novel perspective to tackle problems frequently encountered in the deployment of neural networks.

## References

- [1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283, 2018.
- [2] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.
- [3] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.
- [4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.
- [5] Cheng Chen, Xilin Zhang, Yizhou Wang, and Fang Fang. Measuring the attentional effect of the bottom-up saliency map of natural images. In *International Conference on Intelligent Science and Intelligent Data Engineering*, pages 539–548. Springer, 2012.
- [6] Fang Fang and Yizhou Wang. Image understanding, attention and human early visual cortex. *Frontiers of Electrical and Electronic Engineering*, 7(1):85–93, 2012.
- [7] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- [8] Yiwen Guo, Chao Zhang, Changshui Zhang, and Yurong Chen. Sparse dnns with improved adversarial robustness. In *Advances in neural information processing systems*, pages 242–251, 2018.
- [9] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.
- [10] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.
- [11] 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*, pages 770–778, 2016.
- [12] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
- [13] Weiwei Hu and Ying Tan. Generating adversarial malware examples for black-box attacks based on gan. *arXiv preprint arXiv:1702.05983*, 2017.
- [14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *2015 ICLR*, arXiv preprint arXiv:1412.6980, 2015.
- [15] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *2017 ICLR*, arXiv preprint arXiv:1611.01236, 2017.
- [16] Ji Lin, Chuang Gan, and Song Han. Defensive quantization: When efficiency meets robustness. *arXiv preprint arXiv:1904.08444*, 2019.
- [17] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.
- [18] 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.
- [19] Eric Mandelbaum. Associationist theories of thought. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2017 edition, 2017.

- [20] 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.
- [21] 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*, pages 506–519. ACM, 2017.
- [22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [23] Dawn Song, Kevin Eykholt, Ivan Evtimov, Earlene Fernandes, Bo Li, Amir Rahmati, Florian Tramer, Atul Prakash, and Tadayoshi Kohno. Physical adversarial examples for object detectors. In *12th USENIX Workshop on Offensive Technologies (WOOT 18)*, 2018.
- [24] 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.
- [25] The Tesla team. A tragic loss. <https://www.tesla.com/blog/tragic-loss>. 30.06.2016.
- [26] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, pages 2074–2082, 2016.
- [27] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.
- [28] Kaidi Xu, Sijia Liu, Pu Zhao, Pin-Yu Chen, Huan Zhang, Quanfu Fan, Deniz Erdogmus, Yanzhi Wang, and Xue Lin. Structured adversarial attack: Towards general implementation and better interpretability. In *International Conference on Learning Representations*, 2019.
- [29] Xilin Zhang, Li Zhaoping, Tiangang Zhou, and Fang Fang. Neural activities in v1 create a bottom-up saliency map. *Neuron*, 73(1):183–192, 2012.
- [30] Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044*, 2017.