Adversarial examples in deep learning

Grégory Châtel

Disaitek
Intel Software Innovator

@rodgzilla
github.com/rodgzilla

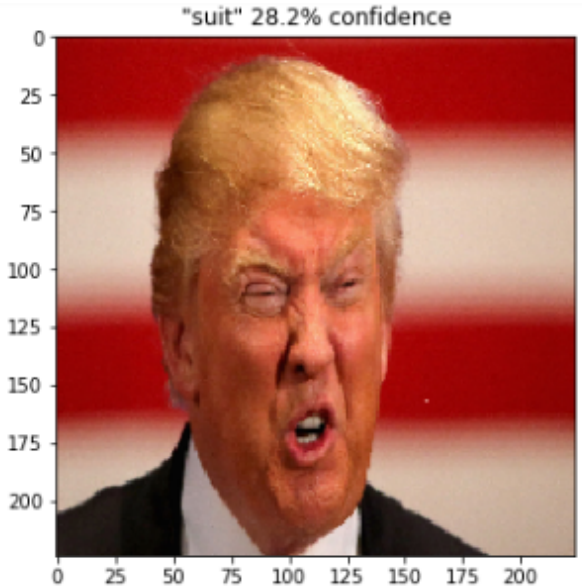
05/08/2018

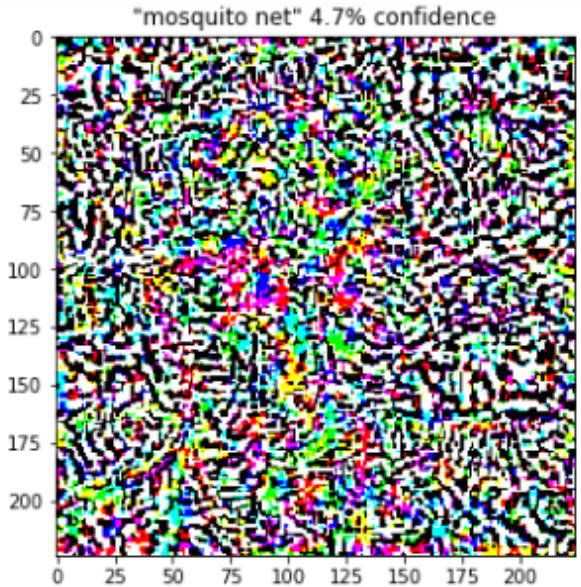
What is an adversarial example?

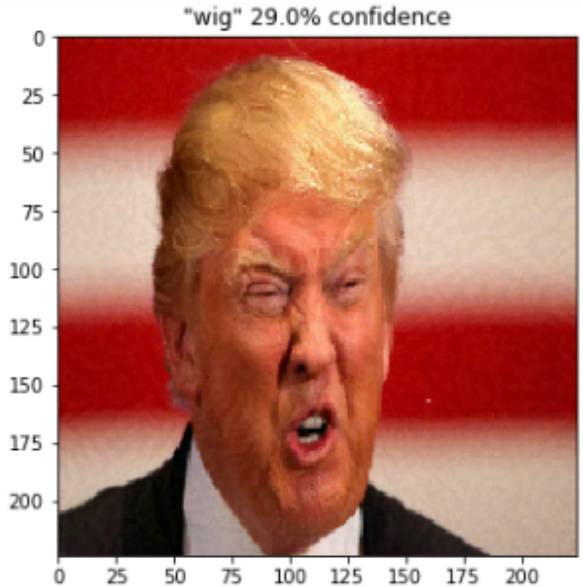
An *adversarial example* is a sample of input data which has been modified *very slightly* in a way that is intended to cause a machine learning classifier to misclassify it.

What is an adversarial example?

An *adversarial example* is a sample of input data which has been modified *very slightly* in a way that is intended to cause a machine learning classifier to misclassify it.

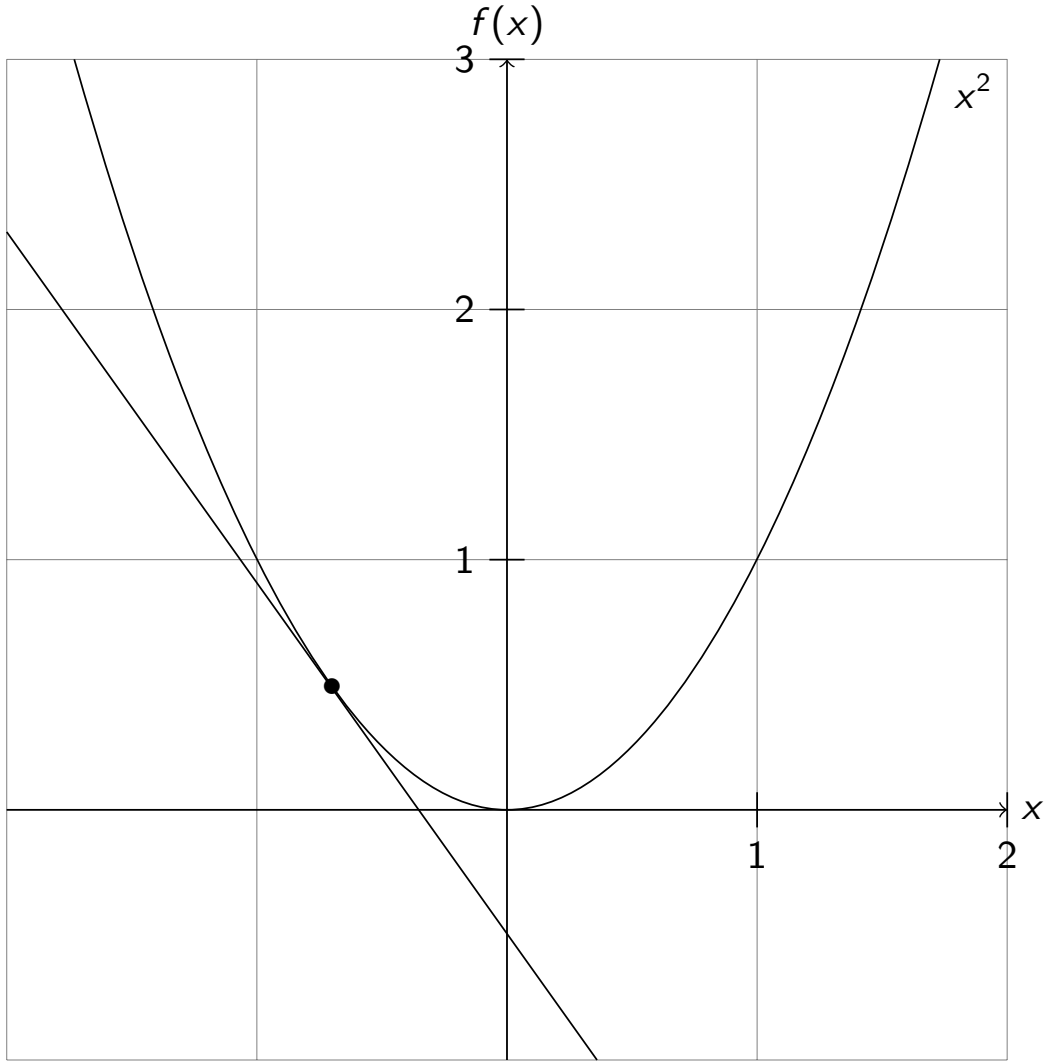






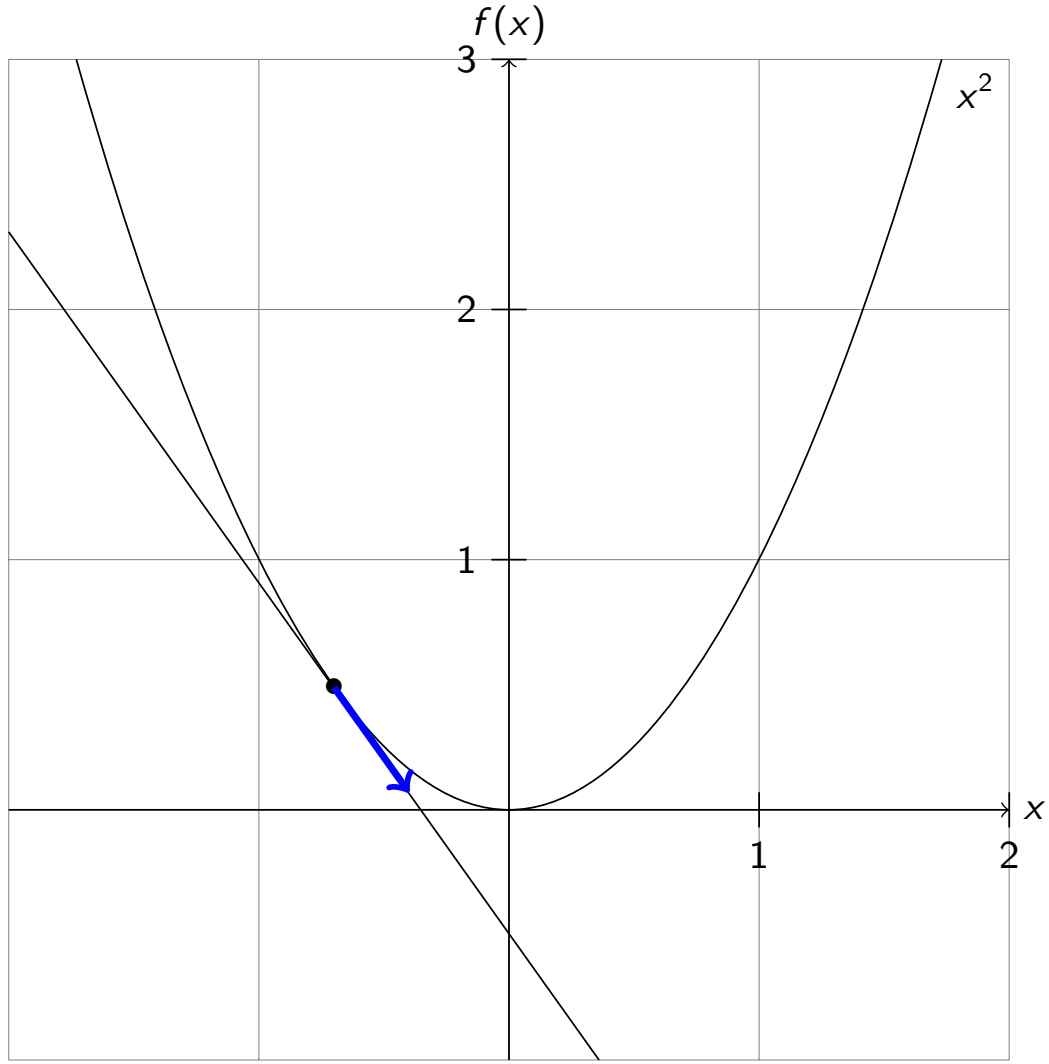
Gradient descent

Basic concept



Gradient descent

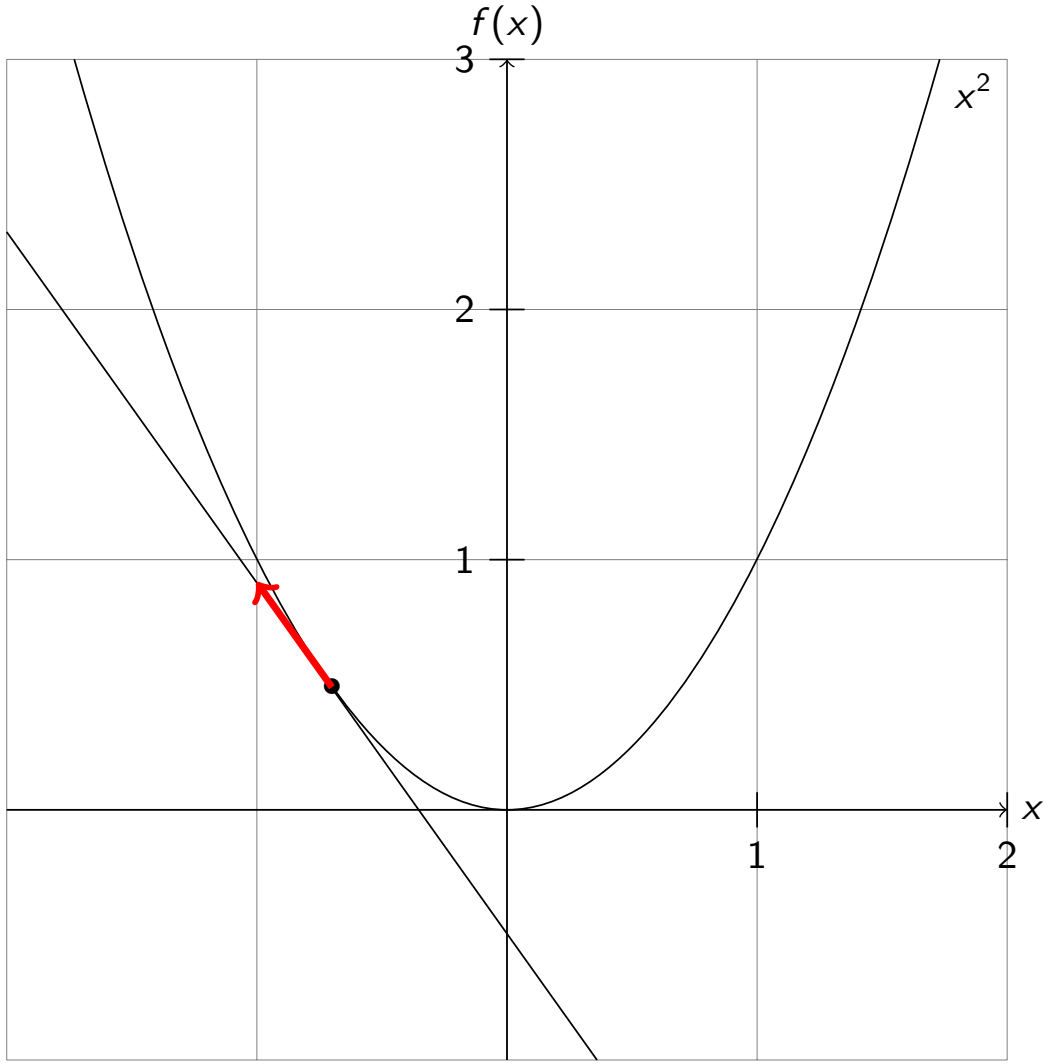
Basic concept



Optimization

Gradient descent

Basic concept



De-optimization

Neural networks

To train and evaluate neural networks, we use a *loss function* $L(\theta, x, y)$ with θ the parameters of the models, x an input and y the real value corresponding to x. This function measures *how good* the prediction of the model is on a specific example.

Neural networks

To train and evaluate neural networks, we use a *loss function* $L(\theta, x, y)$ with θ the parameters of the models, x an input and y the real value corresponding to x. This function measures *how good* the prediction of the model is on a specific example.

To train a neural network we compute the derivative of L according to the weights (θ) and use the result to update θ in order to decrease the loss value.

Neural networks

To train and evaluate neural networks, we use a *loss function* $L(\theta, x, y)$ with θ the parameters of the models, x an input and y the real value corresponding to x. This function measures *how good* the prediction of the model is on a specific example.

To train a neural network we compute the derivative of L according to the weights (θ) and use the result to update θ in order to decrease the loss value.

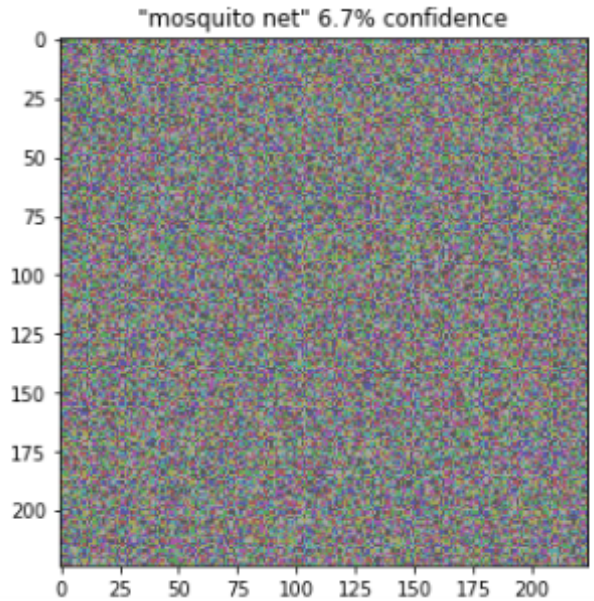
To create an adversarial sample, we compute the derivative of L according to the input (x) and use the result to update the pixel values of x in order to increase the loss value. It happens that such modifications of x often cause the network to misclassify it.

Random noise perturbation

Do we really need to do that?

Random noise perturbation

Yep



Fast Gradient Sign Method [2015]

Let x be the original image, θ the parameters of the model, y the target associated with x and $L(\theta, x, y)$ the loss function.

We compute the gradient of the loss function according to the input pixels.

$$\nabla_x L(\theta, x, y)$$

The perturbation is the signs of these derivatives multiplied by a small number ε.

$$\eta = \varepsilon \ \text{sign}(\nabla_x L(\theta, x, y))$$

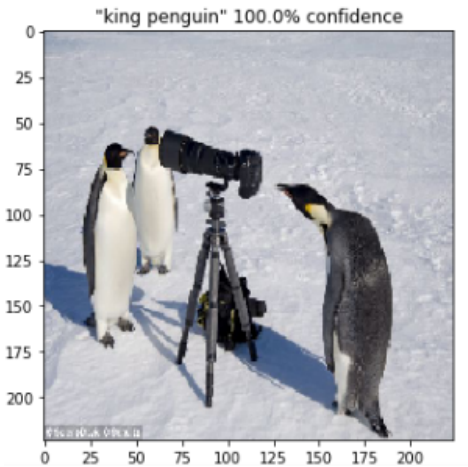
The final adversarial sample is the sum of the original image and the perturbation.

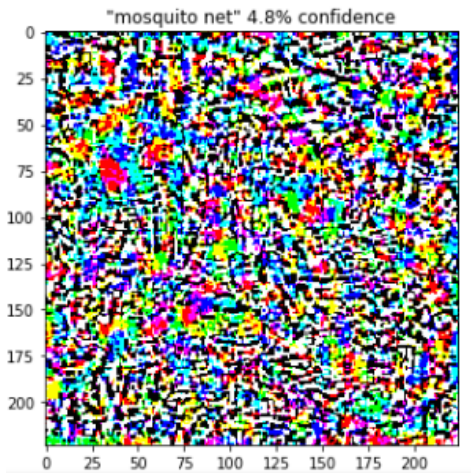
$$x_{adv} = x + \eta$$

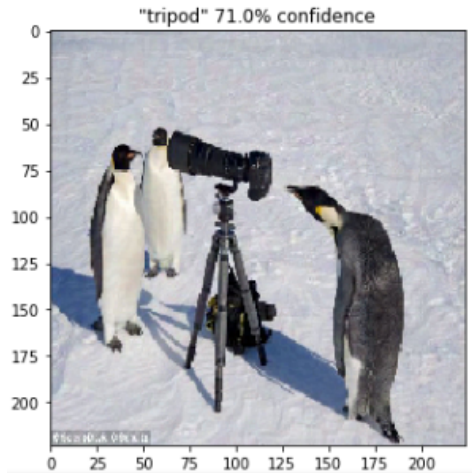
Fast Gradient Sign Method

VGG16 network

$$x + \varepsilon \operatorname{sign}(\nabla_x L(\theta, x, y)) = x_{\text{adv}}$$







Practical black-box attacks against deep learning systems using adversarial examples [2016]

or good luck getting gradients out of your self-driving car

We can train a new model M' to solve the same classification task as the target model M.

We can train a new model M' to solve the same classification task as the target model M.

Once trained, we can create an adversarial sample x'_{adv} for the M' model and experiences have shown that x'_{adv} will also fool M very often.

We can train a new model M' to solve the same classification task as the target model M.

Once trained, we can create an adversarial sample x'_{adv} for the M' model and experiences have shown that x'_{adv} will also fool M very often.

What if we do not have a training set for the target network? Well... build one using M predictions.

We can train a new model M' to solve the same classification task as the target model M.

Once trained, we can create an adversarial sample x'_{adv} for the M' model and experiences have shown that x'_{adv} will also fool M very often.

What if we do not have a training set for the target network? Well... build one using M predictions.

"After labeling 6,400 synthetic inputs to train our substitute (an order of magnitude smaller than the training set used by MetaMind) we find that their DNN misclassifies adversarial examples crafted with our substitute at a rate of 84.24%"

- Papernot et al., about their attack on the MetaMind deep neural network.

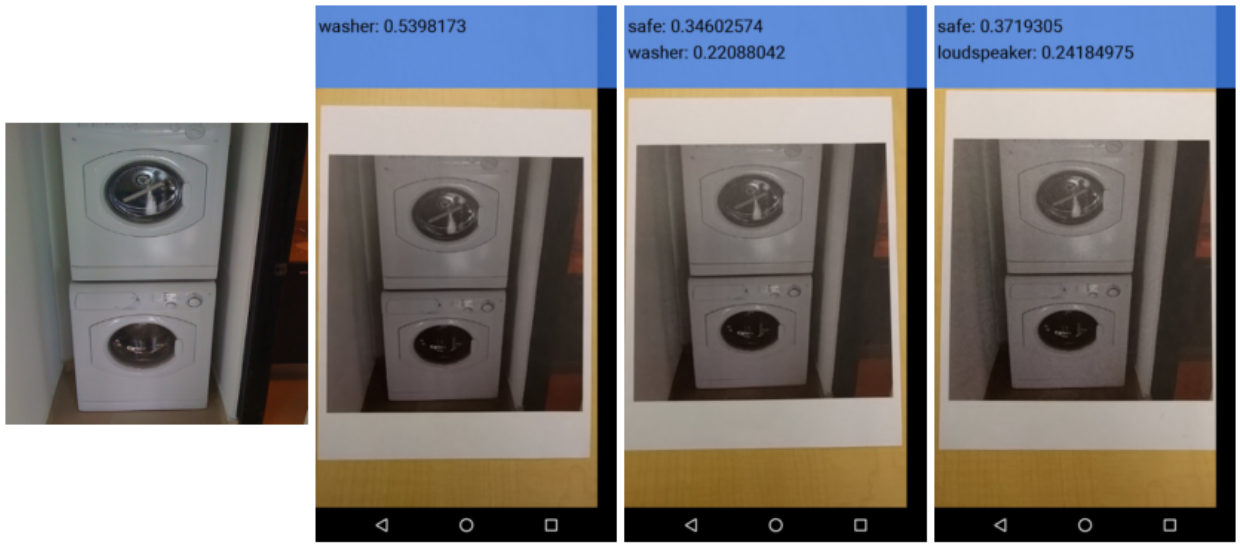
Adversarial examples in the physical world [2017]

or good luck attacking a self-driving car with your USB flash drive

Adversarial examples in the physical world [2017]

In real world scenarios, the target network does not take our image files as input. It acquires the data by the network's system (*e.g.* a camera).

It also works, for free.



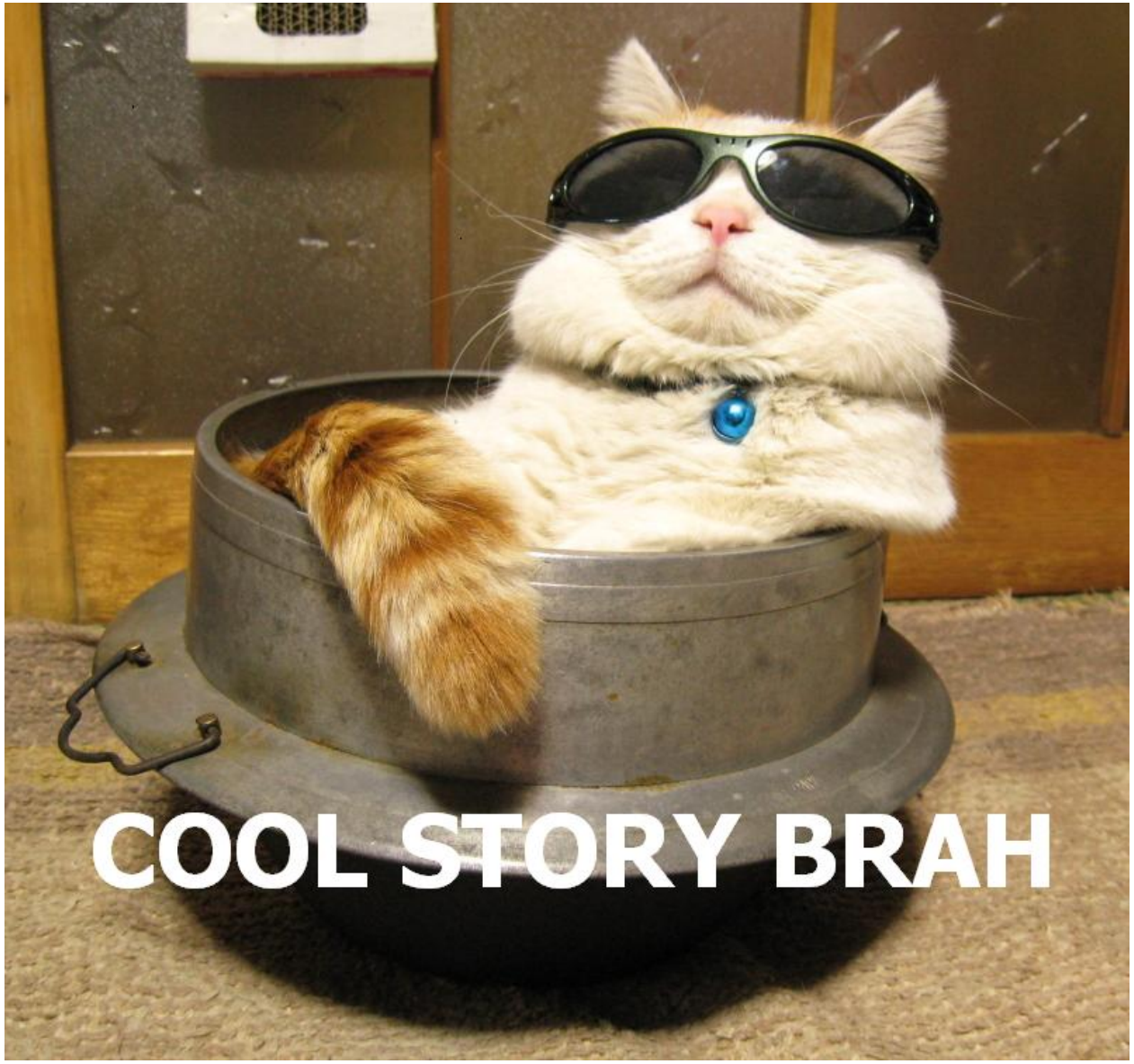
(a) Image from dataset (b) Clean image (c) Adv. image, $\epsilon = 4$ (d) Adv. image, $\epsilon = 8$

"We used images taken from a cell-phone camera as a input to an Inception v3 image classification neural network. We showed that in such a set-up, a significant fraction of adversarial images crafted using the original network are misclassified even when fed to the classifier through the camera."

- Kurakin et al.

Robust Physical-World Attacks on Machine Learning Models [2017]

or good luck perturbating the background in real life



Robust Physical-World Attacks on Machine Learning Models [2017]

Perturbations can also be constrained to mimic vandalism or art in order to go unreported by casual observers.

This concept allows the algorithm to create perturbations with much bigger magnitudes since we do not care about being perceived anymore

Defense

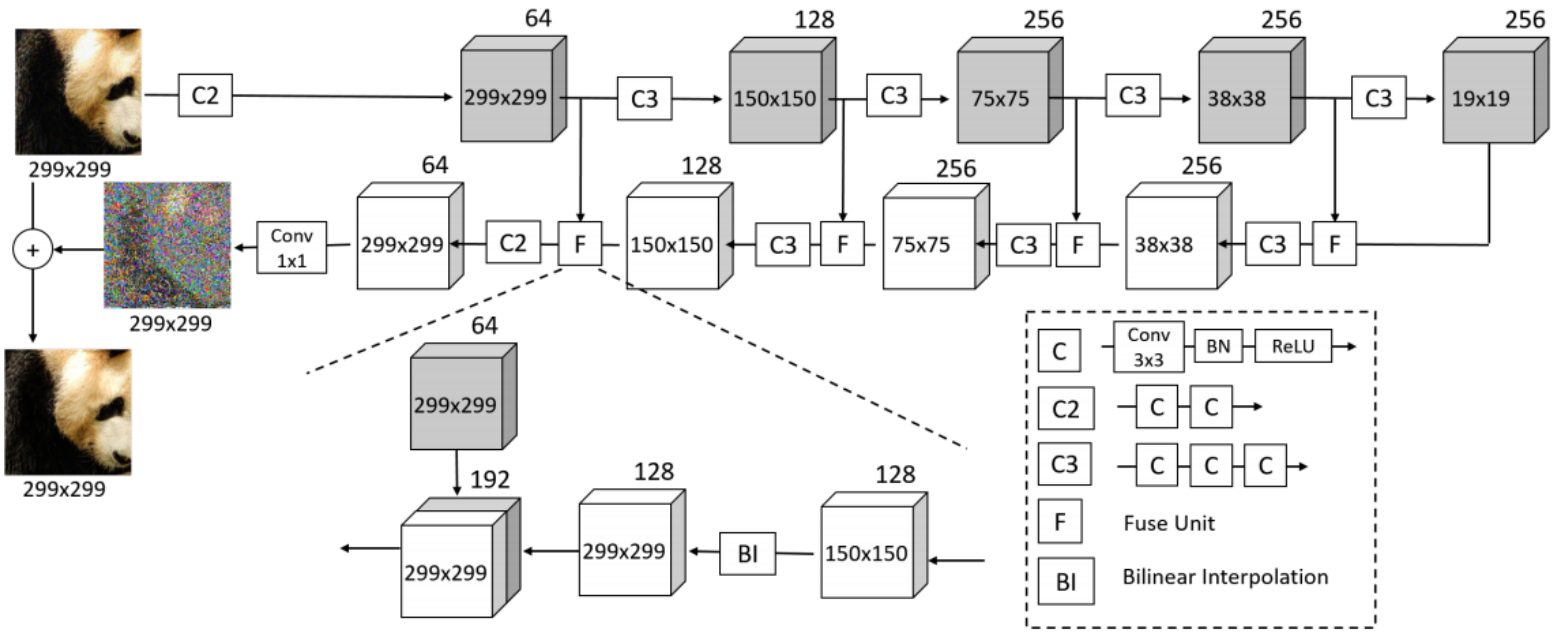
Making a network robust to adversarial perturbation

Black-box defense

Denoising strategies

Train a neural network to remove adversarial perturbation before using it.

The winning team of the defense track of NIPS 2017 competition trained a denoising U-net to remove adversarial noise.



White-box defense

Adversarial training

White-box defense is a much harder problem.

We can generate adversarial samples and train the network to produce the correct classification on these new data points.

- Works well against FGSM
- Expensive in training time
- Tends to overfit the attack used during the training
- Does not defend subtler white-box attacks like RAND+FGSM

Defending machine learning

Still open problem

"No method of defending against adversarial examples is yet completely satisfactory. This remains a rapidly evolving research area."

- Alexey Kurakin, Ian Goodfellow, Samy Bengio et al., March 2018

Use of adversarial examples

What adversarial samples can be used for?

Use of adversarial examples

DARTS: Deceiving Autonomous Cars with Toxic Signs [2018]

Making jokes, OK, but besides this?

Use of adversarial examples

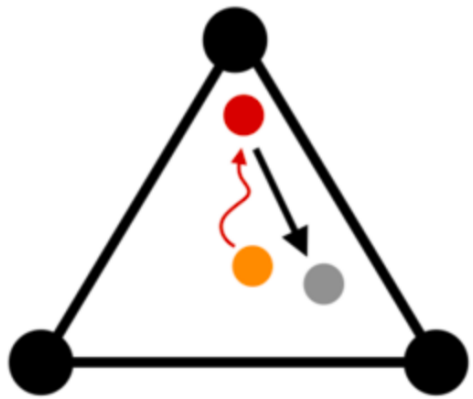
ConvNets and ImageNet Beyond Accuracy: Explanations, Bias Detection, Adversarial Examples and Model Criticism [2017]

Explore the biases of a neural network by analysing the distance of a sample to the decision boundary using adversarial samples.

The distance to the decision boundary is closely related to the magnitude of the perturbation necessary to make a sample cross it.

Use of adversarial examples

Virtual adversarial training: a regularization method for supervised and semi-supervised learning. [2017]



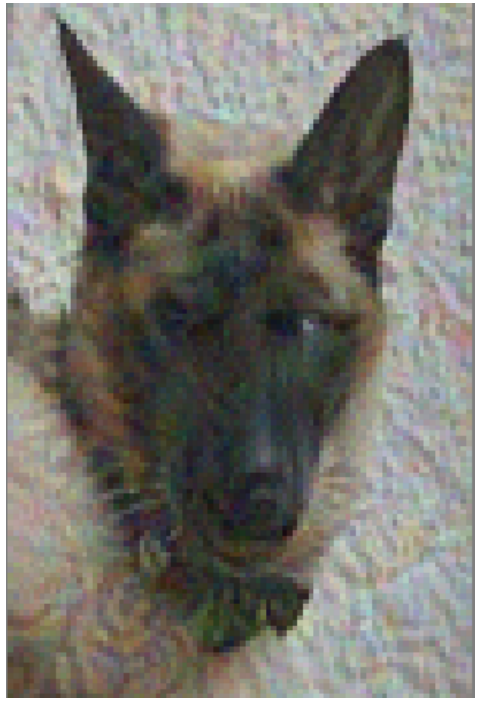
Use unlabelled data to regularize neural network by training it to make the same prediction on a unlabelled image and its adversarial counterpart.

Image from https://thecuriousaicompany.com/mean-teacher/

Adversarial Examples that Fool both Human and Computer Vision [2018]

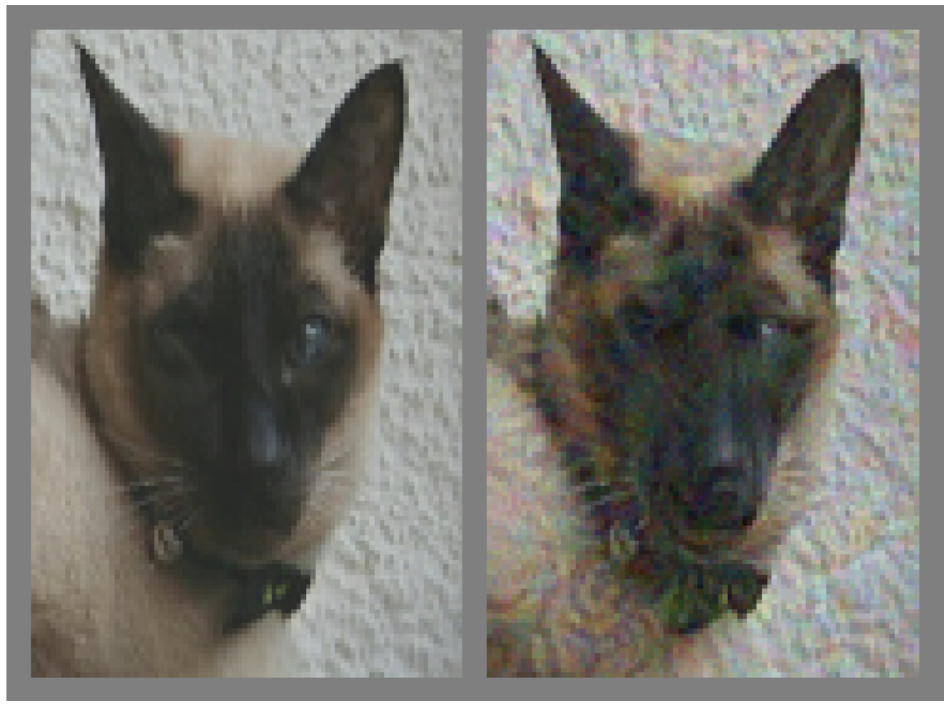
Adversarial Examples that Fool both Human and Computer Vision [2018]

What do you see here?



Adversarial Examples that Fool both Human and Computer Vision [2018]

What do you see here?



By building neural network architecture that closely match the human visual system, the authors have manage to create adversarial samples that fool humans.

References

Research papers:

- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.
- Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2016). Practical black-box attacks against deep learning systems using adversarial examples. arXiv preprint arXiv:1602.02697.
- Kurakin, A., Goodfellow, I., & Bengio, S. (2016). Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533.
- Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., & McDaniel, P. (2017). The Space of Transferable Adversarial Examples. arXiv preprint arXiv:1704.03453.
- Miyato, T., Maeda, S. I., Koyama, M., & Ishii, S. (2017). Virtual adversarial training: a regularization method for supervised and semi-supervised learning. arXiv preprint arXiv:1704.03976.
- Tramèr, F., Kurakin, A., Papernot, N., Boneh, D., & McDaniel, P. (2017). Ensemble Adversarial Training: Attacks and Defenses. arXiv preprint arXiv:1705.07204.
- Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., Rahmati A. & Song, D. (2017). Robust Physical-World Attacks on Machine Learning Models. arXiv preprint arXiv:1707.08945.
- Stock, P., & Cisse, M. (2017). ConvNets and ImageNet Beyond Accuracy: Explanations, Bias Detection, Adversarial Examples and Model Criticism. arXiv preprint arXiv:1711.11443.
- Liao, F., Liang, M., Dong, Y., Pang, T., Zhu, J., & Hu, X. (2017). Defense against Adversarial Attacks Using High-Level Representation Guided Denoiser. arXiv preprint arXiv:1712.02976.
- Sitawarin, C., Bhagoji, A. N., Mosenia, A., Chiang, M., & Mittal, P. (2018). DARTS: Deceiving Autonomous Cars with Toxic Signs. arXiv preprint arXiv:1802.06430.
- Elsayed, G. F., Shankar, S., Cheung, B., Papernot, N., Kurakin, A., Goodfellow, I., & Sohl-Dickstein, J. (2018). Adversarial Examples that Fool both Human and Computer Vision. arXiv preprint arXiv:1802.08195.
- Kurakin, A., Goodfellow, I., Bengio, S., Dong, Y., Liao, F., Liang, M., ... & Wang, J. (2018). Adversarial Attacks and Defences Competition. arXiv preprint arXiv:1804.00097.

Implementations:

- github.com/tensorflow/cleverhans
- github.com/rodgzilla/machine_learning_adversarial_examples

Targeted perturbation

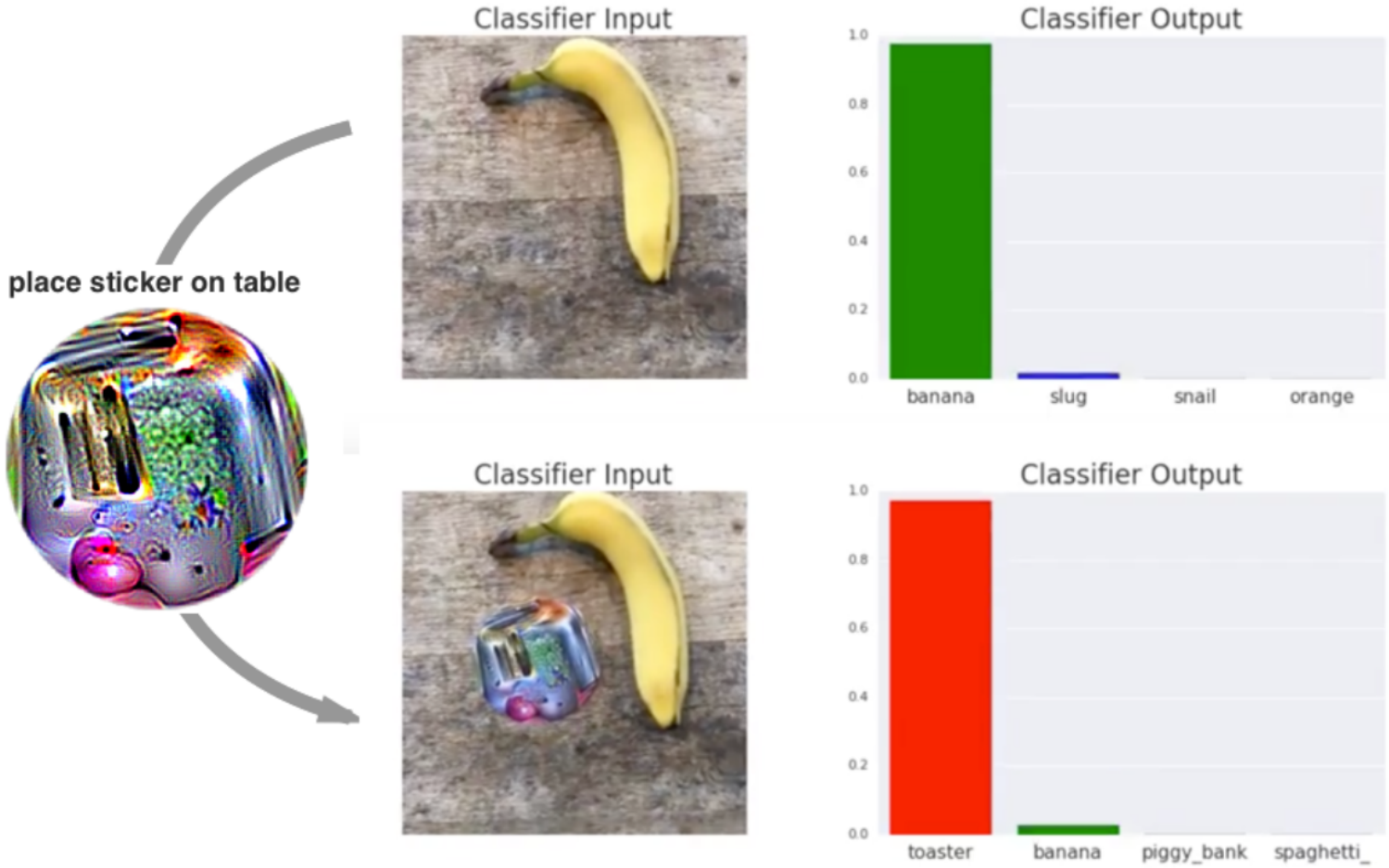
Papernot algorithm

This algorithm iteratively computes the adversarial saliency value $S(x,t)[i]$ of pixel i of the image x according to the class t.

$$S(x,t)[i] = \begin{cases} 0 \quad \text{if} \quad \frac{\mathrm{d}L_t}{\mathrm{d}x_i}(x) < 0 \quad \text{or} \quad \sum_{j \neq t} \frac{\mathrm{d}L_j}{\mathrm{d}x_i}(x) > 0 \\ \\ \frac{\mathrm{d}L_t}{\mathrm{d}x_i}(x) \left| \sum_{j \neq t} \frac{\mathrm{d}L_j}{\mathrm{d}x_i}(x) \right| \text{ otherwise.} \end{cases}$$

and use it iteratively to produce x_{adv} classified as t by the network.

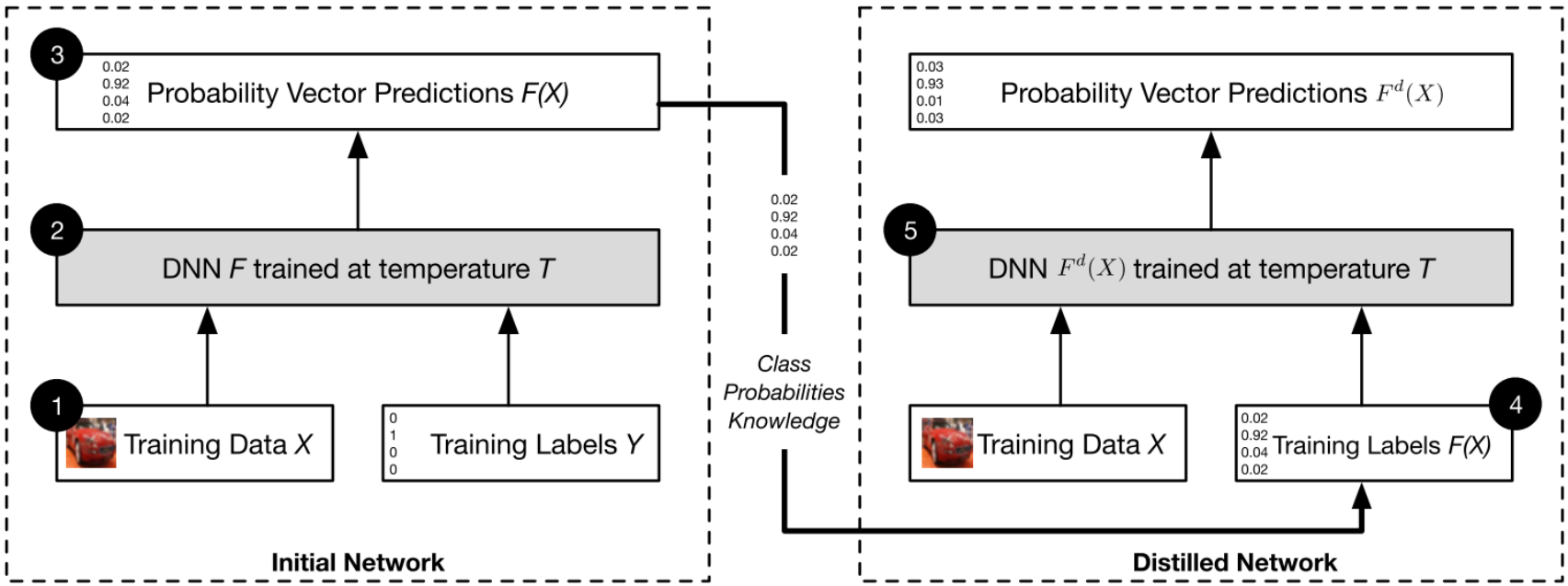
Adversarial patch



Real world adversarial attack on VGG16 using a sticker.

Brown, T. B., Mané, D., Roy, A., Abadi, M., & Gilmer, J. (2017). Adversarial patch. arXiv preprint arXiv:1712.09665.

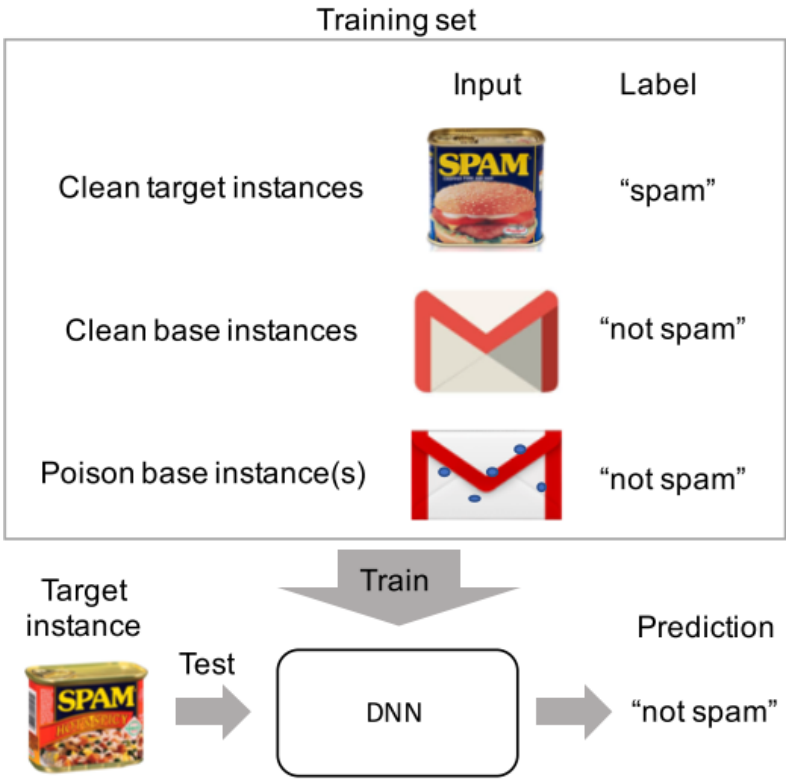
Defensive distillation

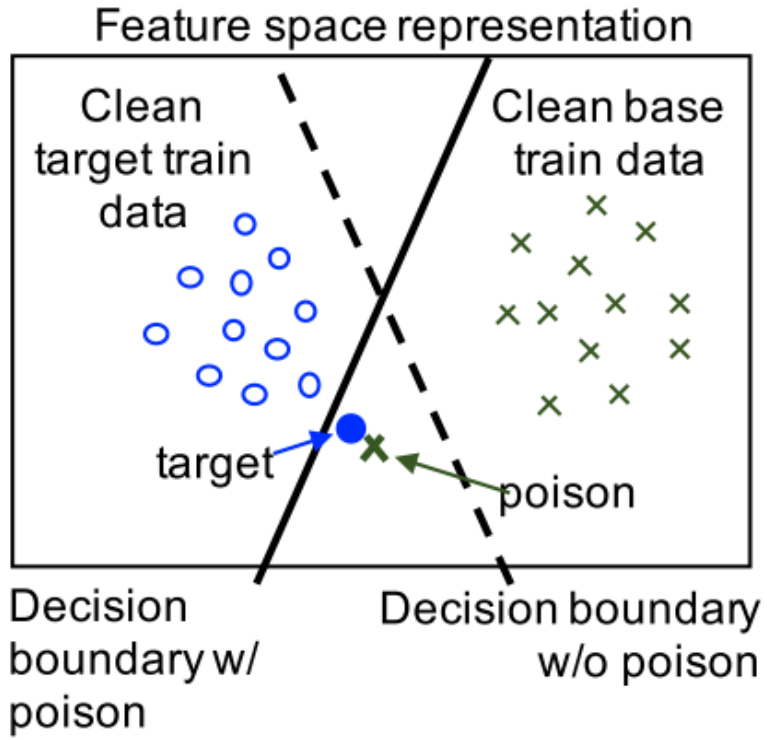


Training a network with explicit relative information about classes prevents models from fitting too tightly to the data.

Papernot, N., McDaniel, P., Wu, X., Jha, S., & Swami, A. (2016). Distillation as a defense to adversarial perturbations against deep neural networks. In Security and Privacy (SP), 2016 IEEE Symposium on (pp. 582-597). IEEE.

Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks



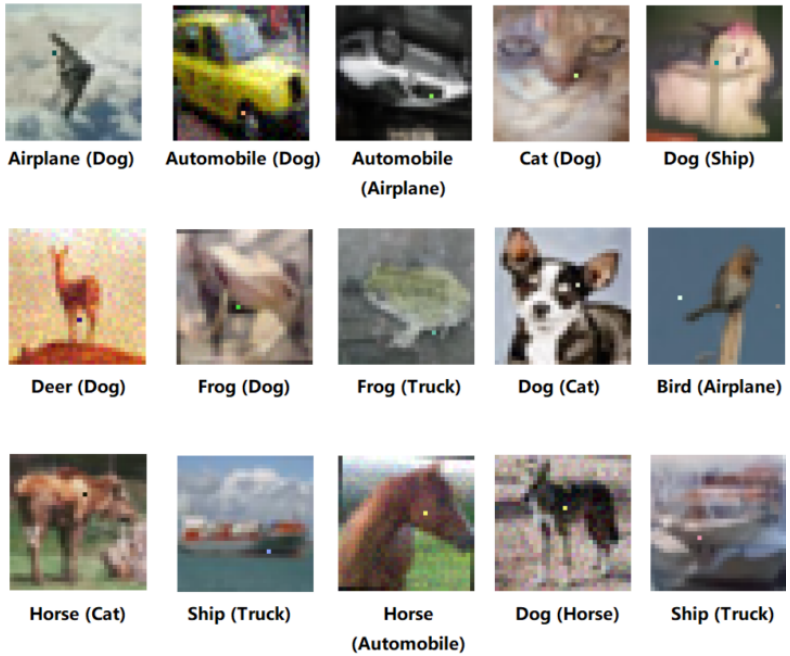


Adversarial examples are a "test time" attack whereas poison frogs are a training time attack.

Perturbed examples are added to the training data of a neural network to obtain a specific behavior at training time.

Shafahi, A., Huang, W. R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., & Goldstein, T. (2018). Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks. arXiv preprint arXiv:1804.00792.

One pixel attack for fooling deep neural networks



This paper presents another attack scenario: the attacker is only allowed to modify the value of 1 pixel as much as he wants.

"The results show that 70.97% of the natural images can be perturbed to at least one target class by modifying just one pixel with 97.47% confidence on average."

Su, J., Vargas, D. V., & Kouichi, S. (2017). One pixel attack for fooling deep neural networks. arXiv preprint arXiv:1710.08864.