Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks

Author: Nicolas Papernot, Patrick McDaniel, Xi Wu,
Somesh Jha, and Ananthram Swami

Presenter: Gang Su

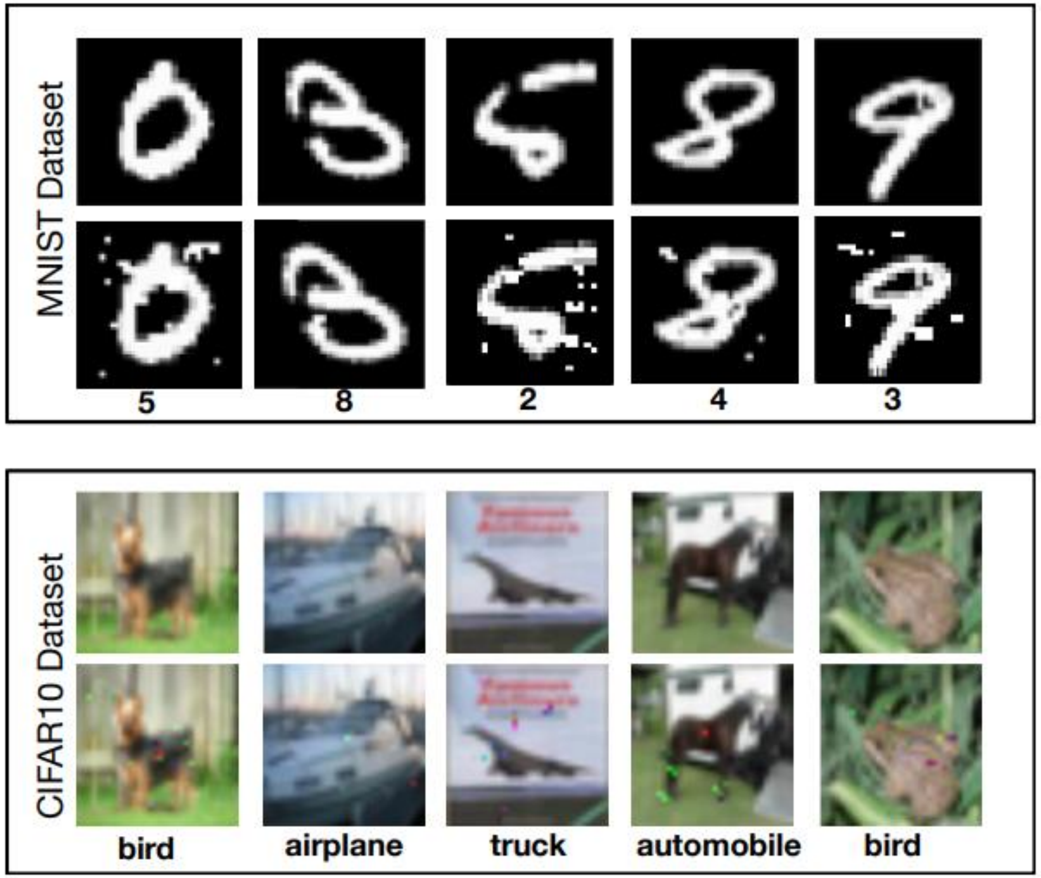
UNIVERSITY OF GEORGIA

Overview

- Motivation
- Related Work
- Adversarial crafting framework
- The Proposed Approach
- Evaluation

Motivation

- Deep learning (especially DNNs) has been shown to perform extremely well on many difficult machine learning problems, such as image recognition, speech recognition, natural language processing, and playing games.
- However, researchers have discovered that existing DNNs are vulnerable to attack. This makes it difficult to apply DNNs in security-critical areas.
- Adversarial examples can force DNNs to produce adversarial-selected outputs or misclassification using carefully crafted inputs.
- Thus, adversarial examples must be taken into account when designing security sensitive systems incorporating DNNs.

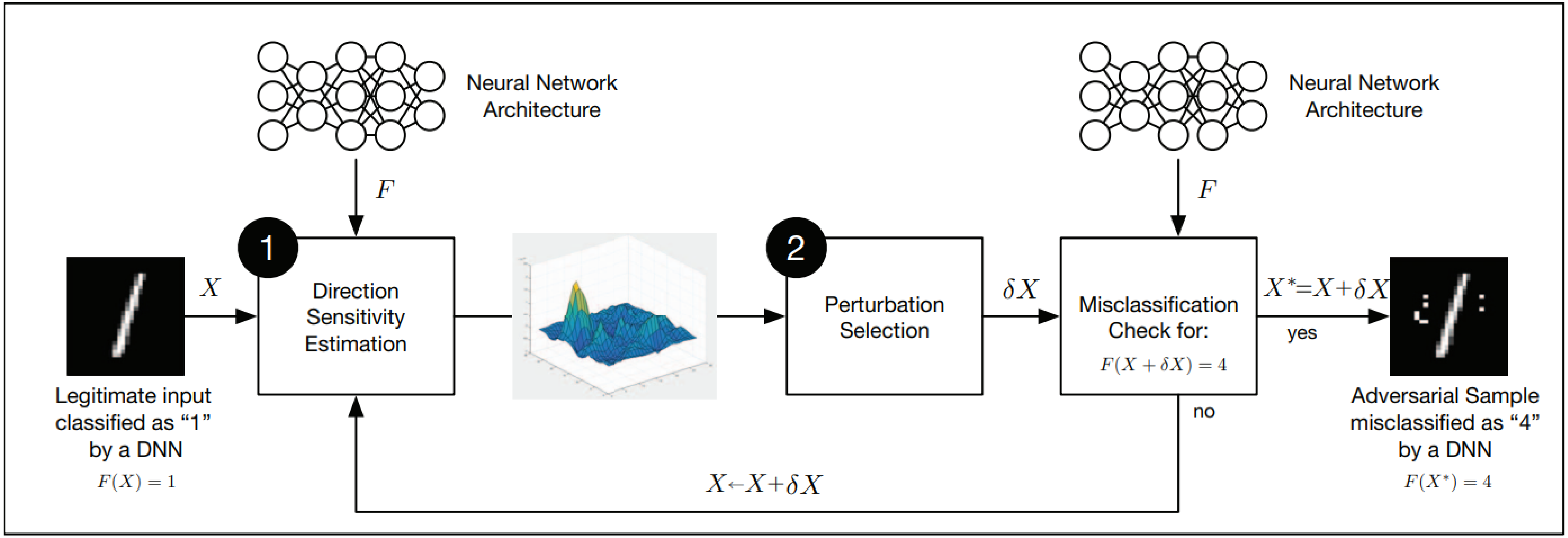


Related Work

- There are very few effective countermeasures available before this paper.
- Goodfellow et al. showed the radial basis activation functions are more resistant to perturbation.
- Gu et al. explored the use of denoising auto-encoders, a DNN type of architecture intended to capture main factors of variation in the data, and showed that they can remove substantial amounts of adversarial noise.
- Previous work considered the problem of constructing such defenses but solutions proposed are deficient in that they require making modifications to the DNN architecture or only partially prevent adversarial samples from being effective.

Adversarial crafting framework

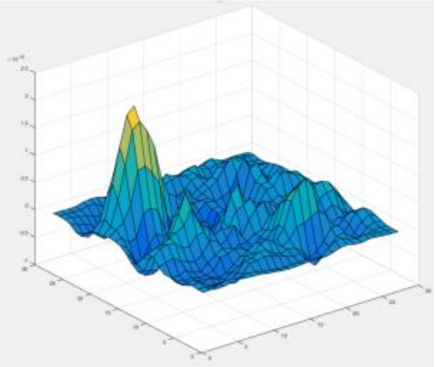
- **Problem: How to effectively defense adversarial examples?**
- How adversarial samples craft?

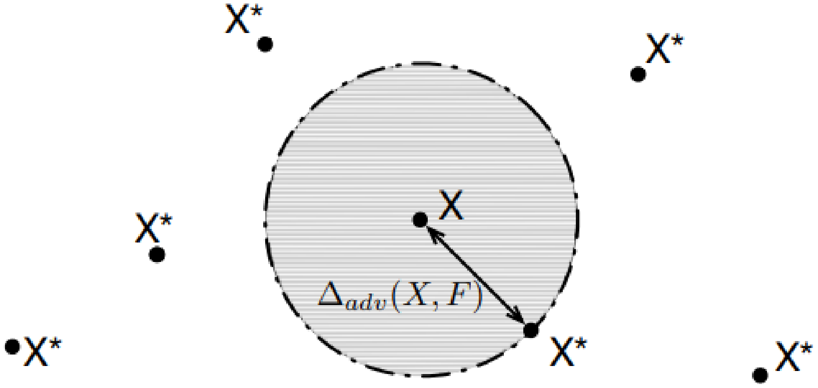


Neural Network Architecture
F
Neural Network Architecture
F
X
1
Direction Sensitivity Estimation
2
Perturbation Selection
δX
Misclassification Check for:
$F(X+\delta X)=4$
$X^*=X+\delta X$
yes
no
Legitimate input classified as "1" by a DNN
$F(X)=1$
Adversarial Sample misclassified as "4" by a DNN
$F(X^*)=4$
$X \leftarrow X+\delta X$

- Goodfellow et al. : Fast Sign Gradient Method (FGSM)
- Pepernot et al. : Jacobian Saliency Map Attack (JSMA)

Adversarial Gradient





$$\Delta_{adv}(X, F) = \arg\min_{\delta X}\{\|\delta X\| : F(X + \delta X) \neq F(X)\}$$

- Attacks based on adversarial samples were primarily exploiting gradients computed to estimate the sensitivity of networks to its input dimensions.

- Refers those gradient as **adversarial gradient**.

- If adversarial gradients are high, crafting adversarial examples becomes easier because small perturbations will induce high network output variations.

- In other word, we must **smooth** the model learned during training by helping the network **generalize** better to samples out side of its training dataset.

The proposed approach – Defensive Distillation

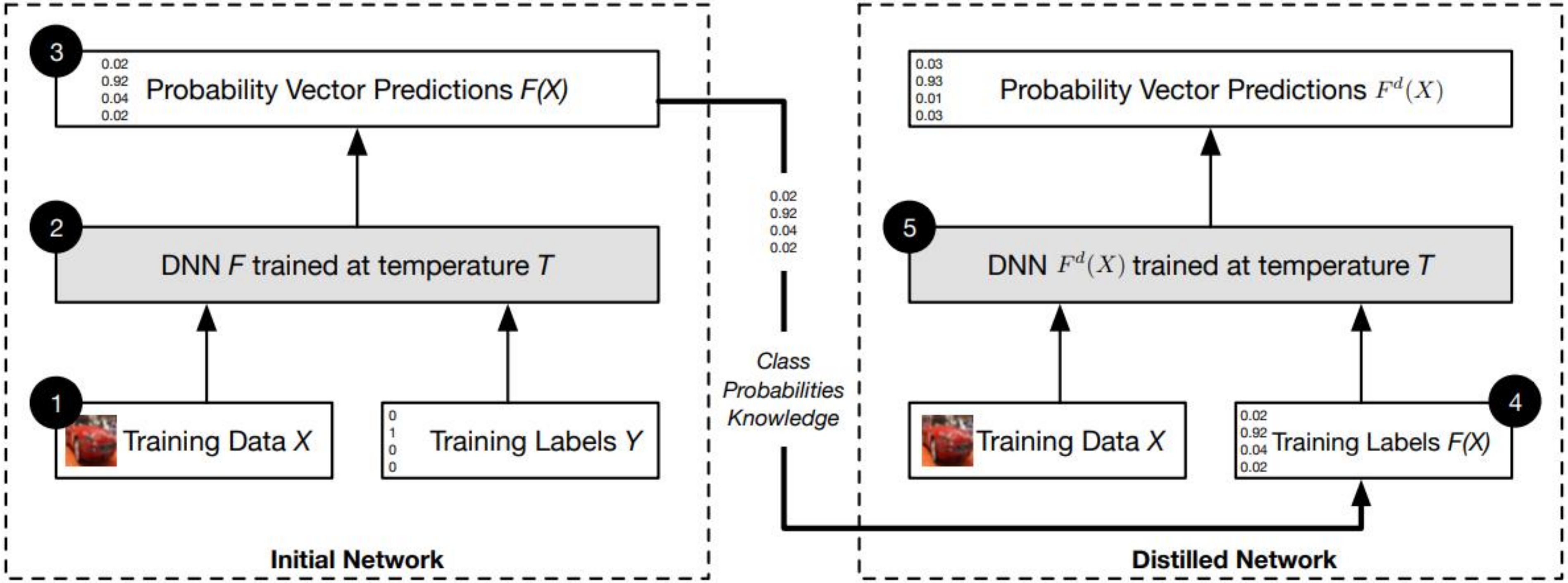


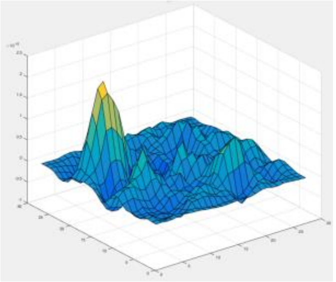
Fig. 5: **An overview of our defense mechanism based on a transfer of knowledge contained in probability vectors through distillation:** We first train an initial network F on data X with a softmax temperature of T. We then use the probability vector $F(X)$, which includes additional knowledge about classes compared to a class label, predicted by network F to train a distilled network F^d at temperature T on the same data X.

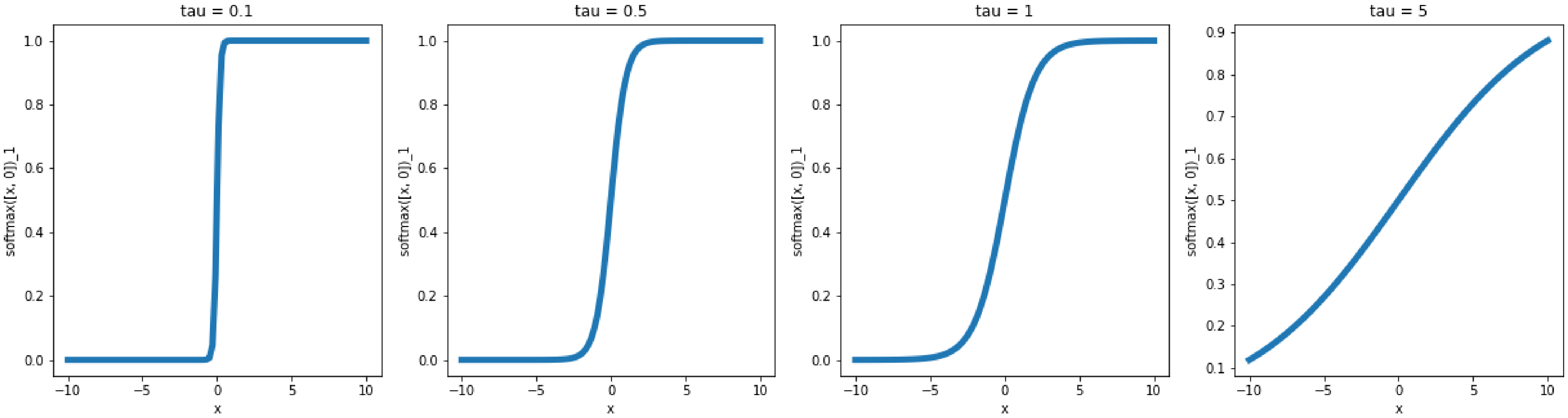
Soft-label: Generalization.
Temperature: Smooth.

Distillation Temperature

$$F(X) = \left[\frac{e^{z_i(X)/T}}{\sum_{l=0}^{N-1} e^{z_l(X)/T}} \right]_{i \in 0..N-1}$$

Smooth





Evaluation

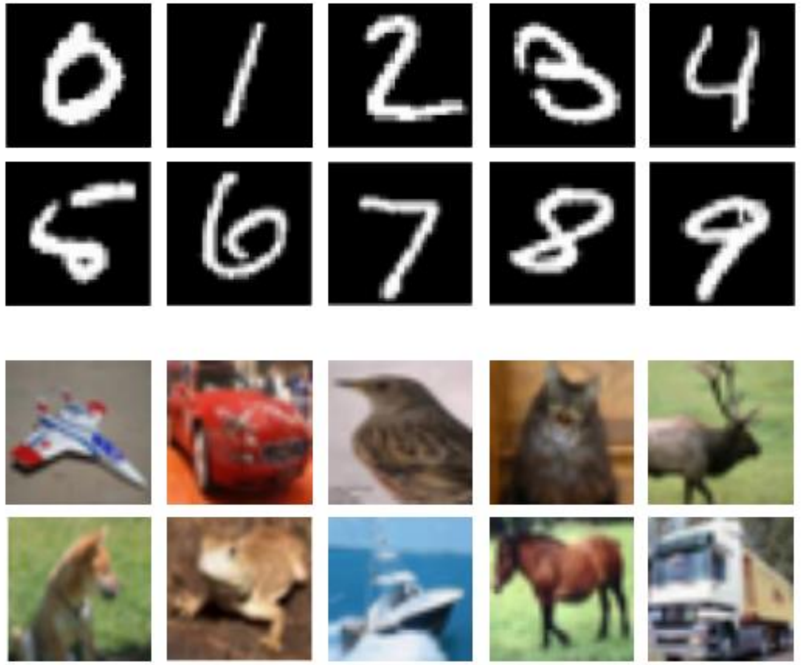
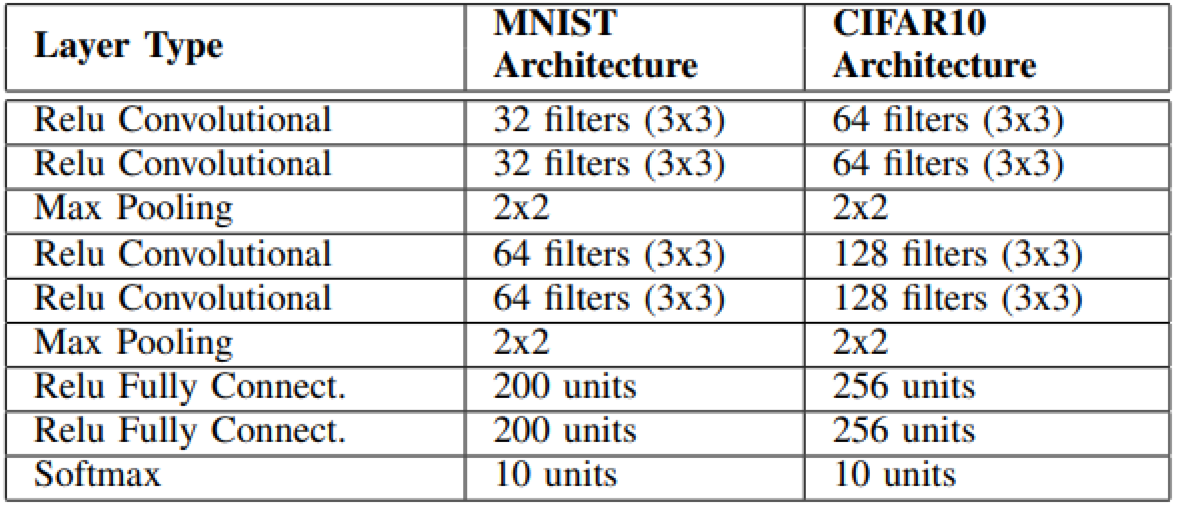
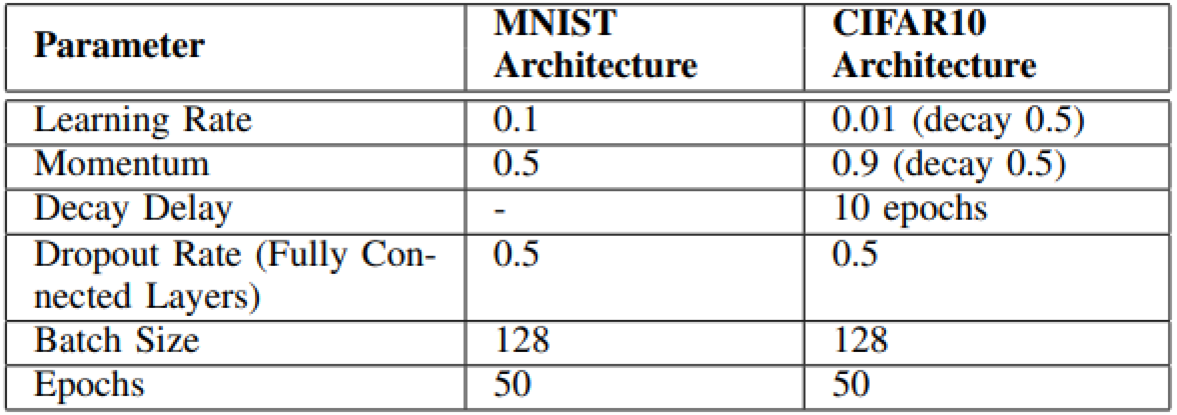


Fig. 6: **Set of legitimate samples:** these samples were extracted from each of the 10 classes of the MNIST handwritten digit dataset (top) and CIFAR10 image dataset (bottom).



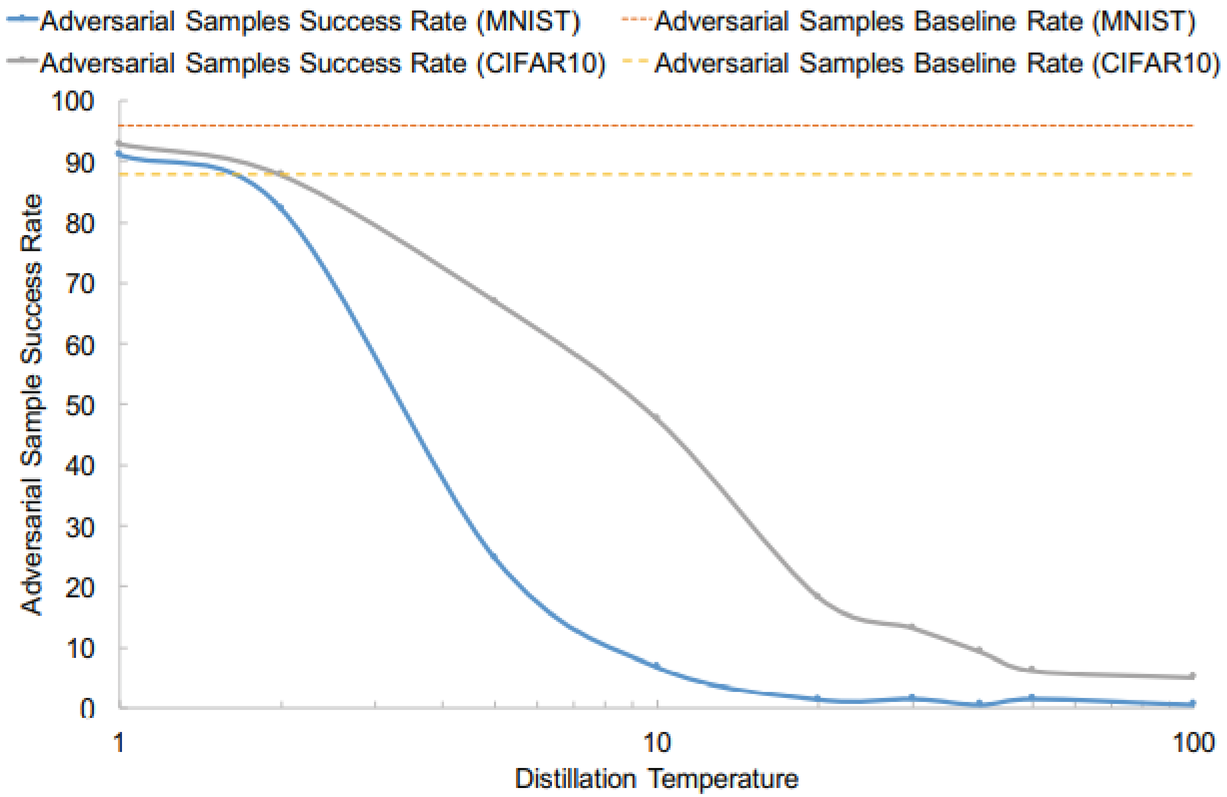
Layer Type	**MNIST Architecture**	**CIFAR10 Architecture**
Relu Convolutional	32 filters (3x3)	64 filters (3x3)
Relu Convolutional	32 filters (3x3)	64 filters (3x3)
Max Pooling	2x2	2x2
Relu Convolutional	64 filters (3x3)	128 filters (3x3)
Relu Convolutional	64 filters (3x3)	128 filters (3x3)
Max Pooling	2x2	2x2
Relu Fully Connect.	200 units	256 units
Relu Fully Connect.	200 units	256 units
Softmax	10 units	10 units

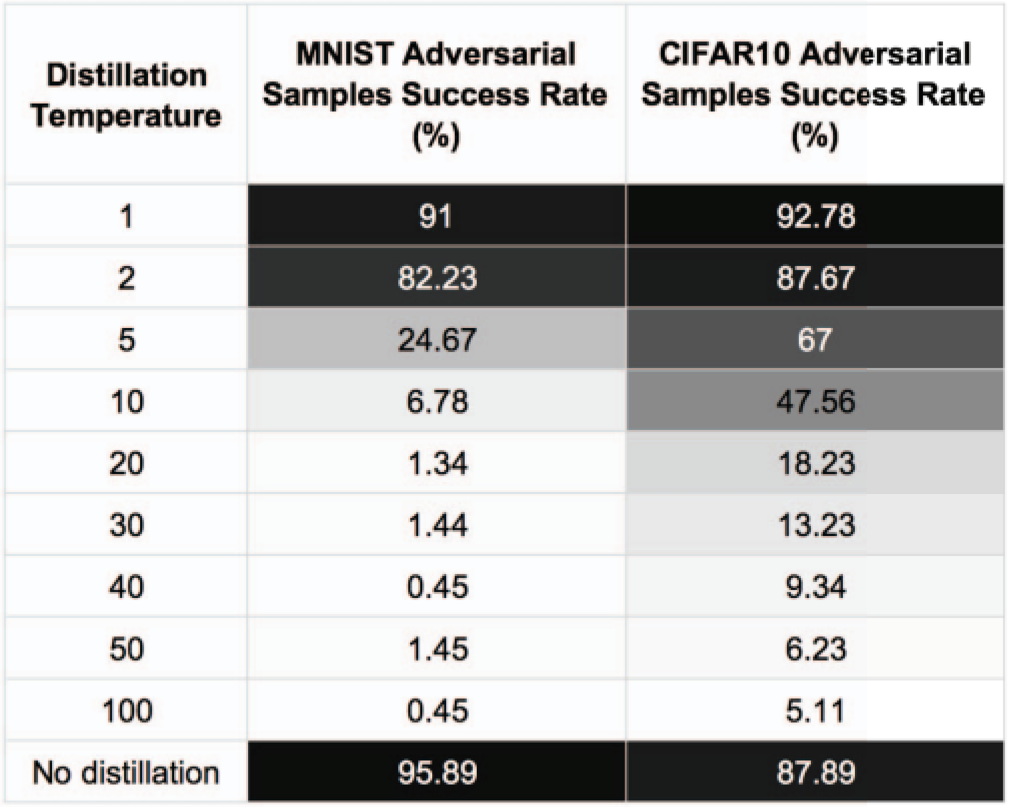


Parameter	**MNIST Architecture**	**CIFAR10 Architecture**
Learning Rate	0.1	0.01 (decay 0.5)
Momentum	0.5	0.9 (decay 0.5)
Decay Delay	-	10 epochs
Dropout Rate (Fully Connected Layers)	0.5	0.5
Batch Size	128	128
Epochs	50	50

Adversarial attack algorithm:
JSMA (Jacobian Saliency Map Attack)

Evaluation





Distillation Temperature	MNIST Adversarial Samples Success Rate (%)	CIFAR10 Adversarial Samples Success Rate (%)
1	91	92.78
2	82.23	87.67
5	24.67	67
10	6.78	47.56
20	1.34	18.23
30	1.44	13.23
40	0.45	9.34
50	1.45	6.23
100	0.45	5.11
No distillation	95.89	87.89

Evaluation

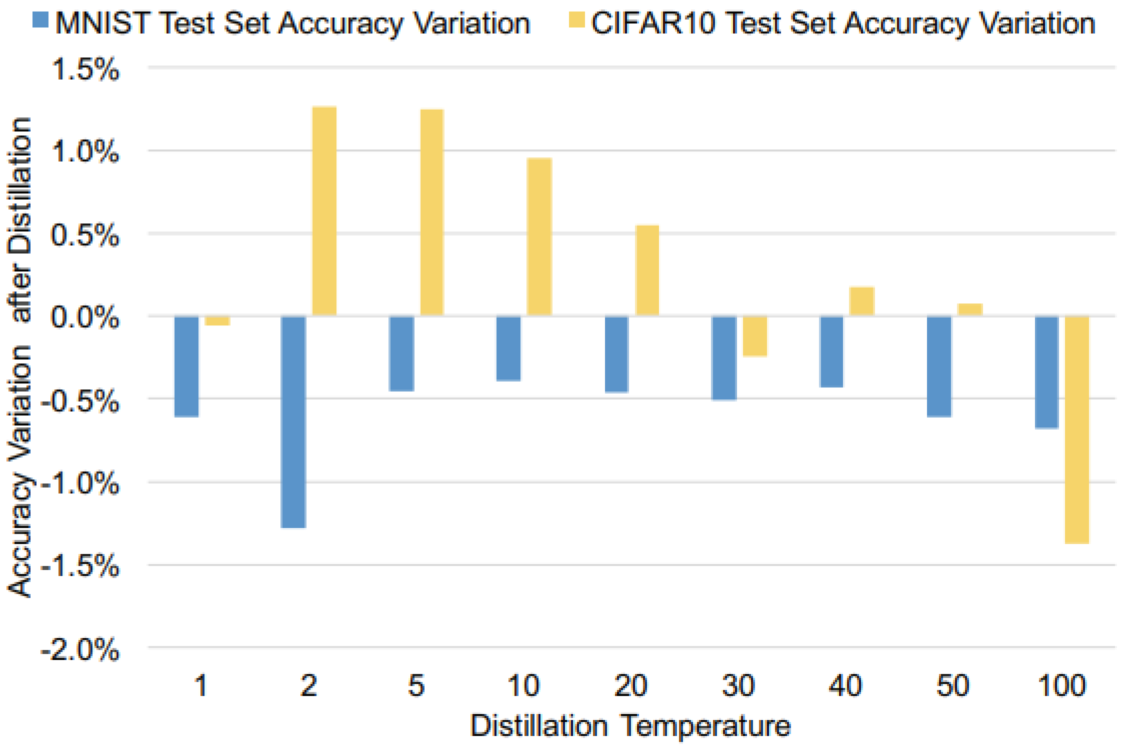


Fig. 8: **Influence of distillation on accuracy:** we plot the accuracy variations of our two architectures for a training with and without defensive distillation. These rates were evaluated on the corresponding test set for various temperature values.

Evaluation

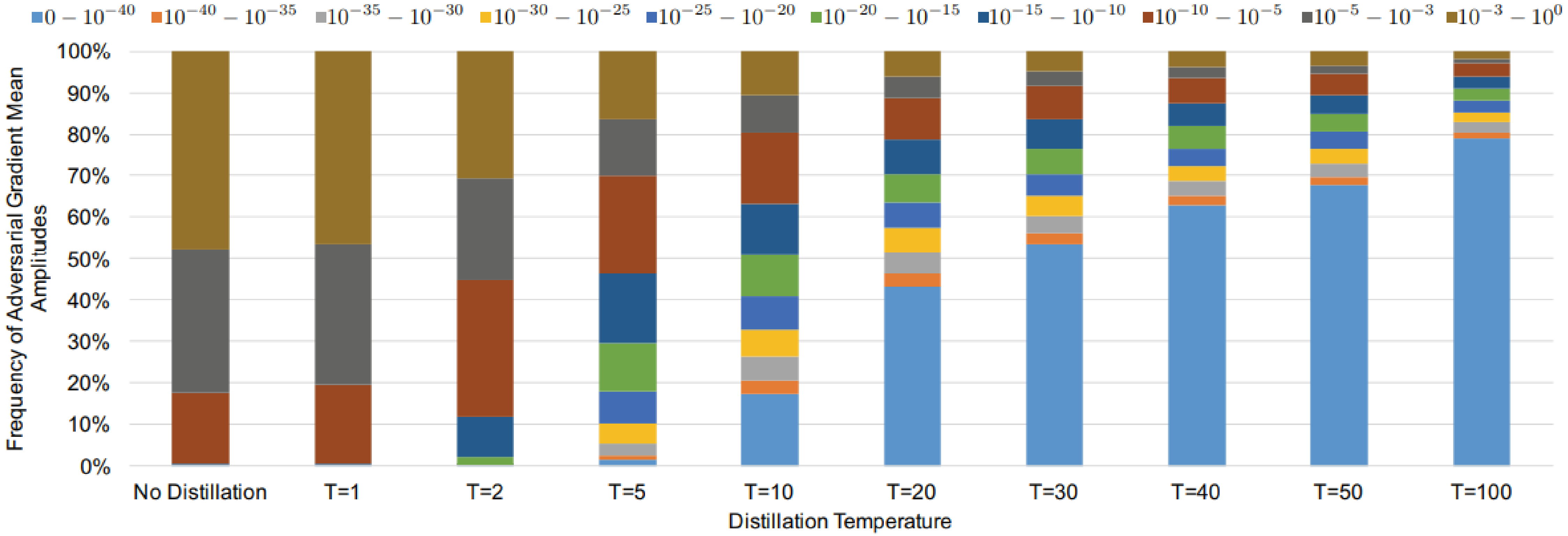


Fig. 9: **An exploration of the impact of temperature on the amplitude of adversarial gradients:** We illustrate how adversarial gradients vanish as distillation is performed at higher temperatures. Indeed, for each temperature considered, we draw the repartition of samples in each of the 10 ranges of mean adversarial gradient amplitudes associated with a distinct color. This data was collected using all 10,000 samples from the CIFAR10 test set on the corresponding DNN model.

Conclusion

- The paper have investigated the use of distillation as a defense against adversarial perturbations.
- 2 important techniques in defensive distillation are **Soft Label** and **Distillation Temperature.**
- The experiment show that defensive distillation can significantly reduce the successfulness of attacks against DNNs without decrease the accuracy.

Towards Evaluating the Robustness of Neural Networks

demonstrates the defensive distillation does not significantly increase the robustness of neural networks by introducing three new attack algorithms (l_0, l_2, l_∞) that are successful on both distilled and undistilled NNs with 100% probability.

Thank you!

Questions?