ICTS, Bengaluru

December 29, 2018

Towards Robust Deep Learning

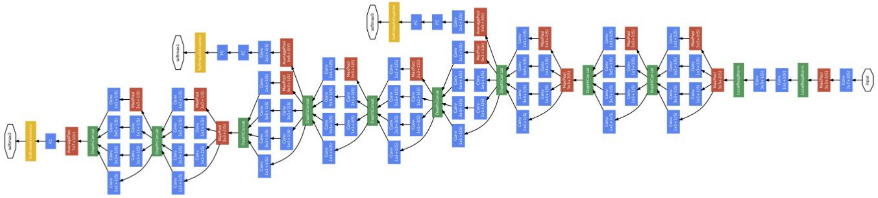
Stephan Zheng

stephan.zheng@salesforce.com
www.stephanzheng.com

salesforce

Deep learning in the lab

$f\left(\right.$  $\left.\right)$



$$= \begin{pmatrix} 0.01 \\ \vdots \\ 0.15 \\ \vdots \\ 0.02 \end{pmatrix} \in \mathbb{R}^n$$

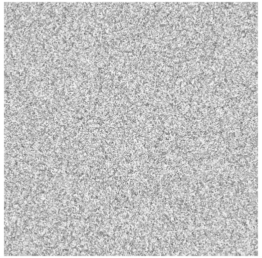
$x \in \mathbb{R}^d$

fawn

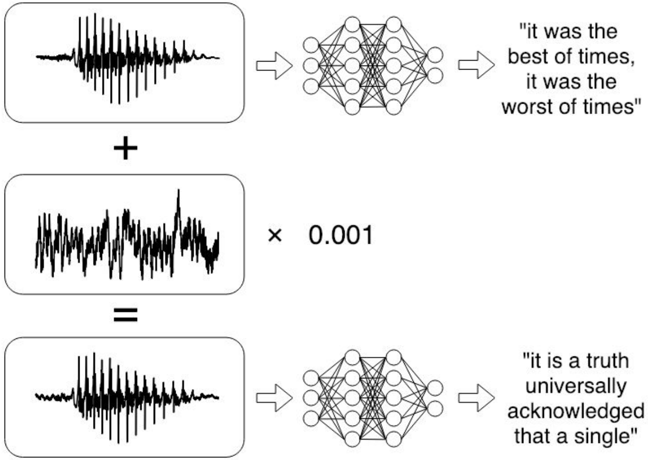
ImageNet Large Scale Visual Recognition Challenge, Russakovsky et al, 2012

Deep learning in the wild

Deep Neural Nets are easily fooled

$f\Big($  $+$  $\Big) = \begin{pmatrix} 0.03 \\ \vdots \\ 0.08 \\ \vdots \\ 0.01 \end{pmatrix} \in \mathbb{R}^n$

monkey



[Carlini et al, 2018]

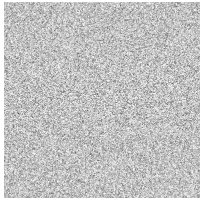
Robustness

$$\forall x' : d(x, x') < \delta \Leftrightarrow D(f(x), f(x')) < \epsilon$$

$f\Big($

$\Big) = \begin{pmatrix} 0.01 \\ \vdots \\ 0.15 \\ \vdots \\ 0.02 \end{pmatrix}$

$f\Big($

$+$

$\Big) = \begin{pmatrix} 0.03 \\ \vdots \\ 0.08 \\ \vdots \\ 0.01 \end{pmatrix}$

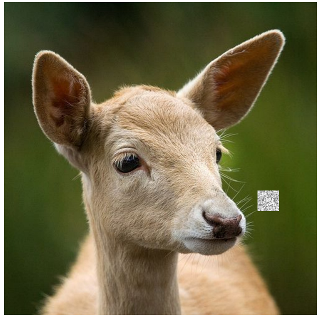
Perturbations $x' = x + dx$ are caused by various sources.

- Natural: compression, cropping, re-scaling
- Crafted (imperceptible) adversarial attacks

Today

Stability Training: robustness against natural perturbations: compression, cropping, re-scaling

Neural Fingerprinting: detecting crafted (imperceptible) adversarial attacks

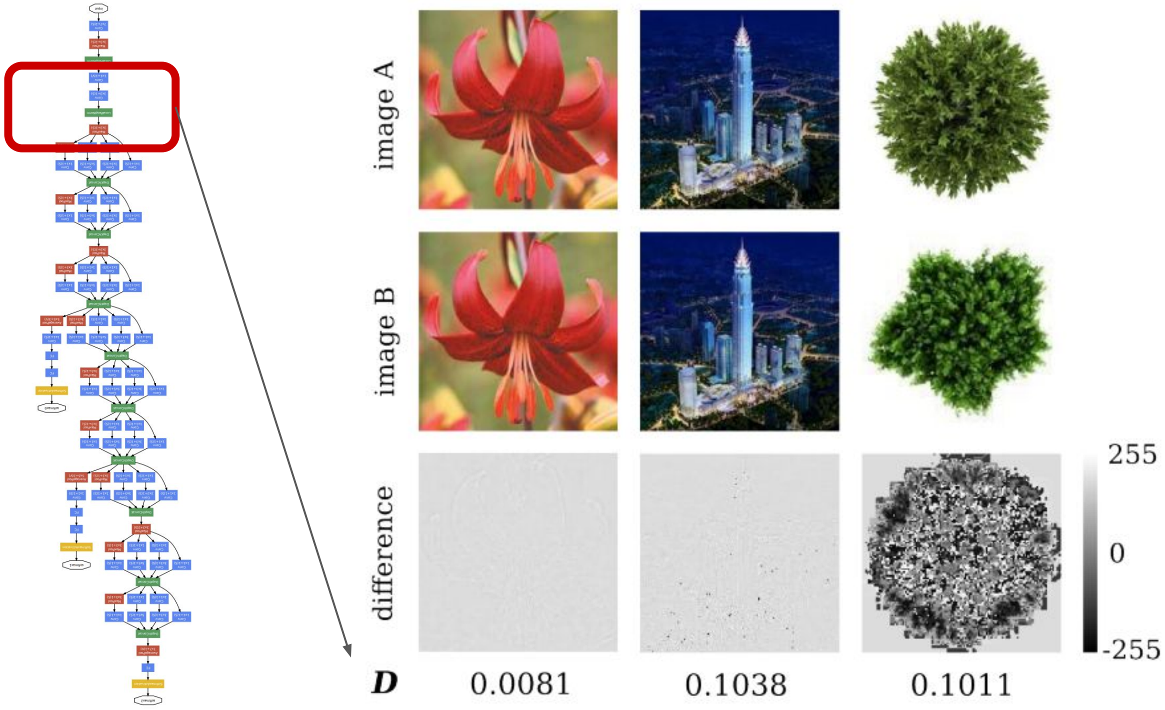


Improving Neural Network Robustness via Stability Training

SZ, Y. Song, T. Leung, I. Goodfellow, CVPR 2016.

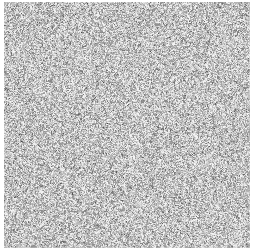
Improving Robustness via Stability Training

GoogleNet (state-of-the-art in 2016) thinks dissimilar pair is more similar than almost identical pair



Approach

Force network to behave similarly on perturbed input, **even if network is wrong**

$f\Big($  $\Big) \approx f\Big($  $+$  $\Big)$

$x \in \mathbb{R}^d$ $\qquad$ $x' \in \mathbb{R}^d$

$$\min_{\theta} \sum_{x'} L_{stab}(x, x'; \theta)$$

$$L_{stab}(x, x'; \theta) = ||f(x) - f(x')||_2^2$$

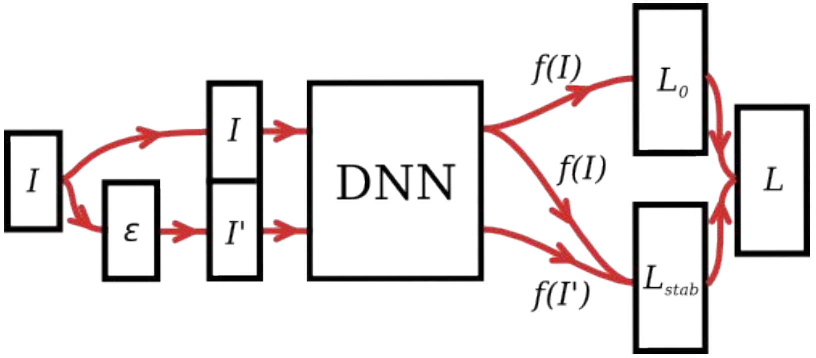
Approach

Force network to behave similarly on perturbed input, **even if network is wrong**

Stochastic data augmentation → during training, $x' = x + dx,\ dx \sim N(0, s^2)$

Simple to implement

Effective although loss surface of typical neural network is highly non-convex

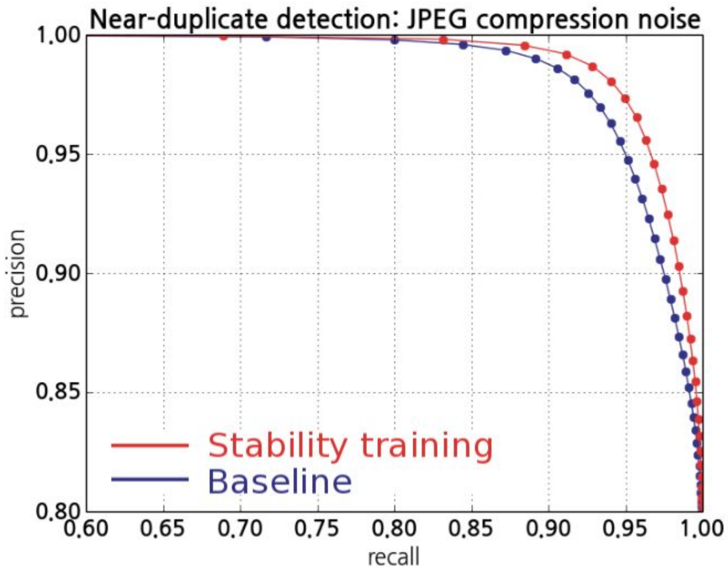


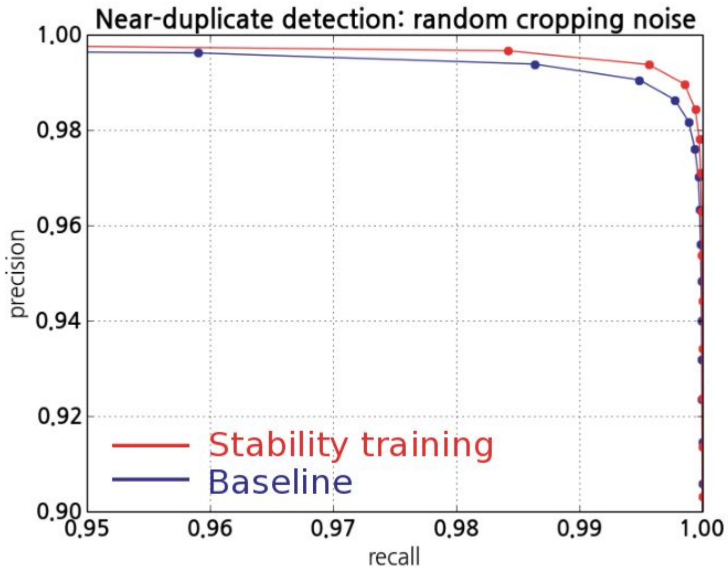
$$L(x, x'; \theta) = L_0(x; \theta) + \alpha L_{stab}(x, x'; \theta)$$
$$L_{stab}(x, x'; \theta) = D(f(x), f(x'))$$

Improving Robustness via Stability Training

Stability training improves **near-duplicate detection** precision-recall by 2-3% on corrupted images.

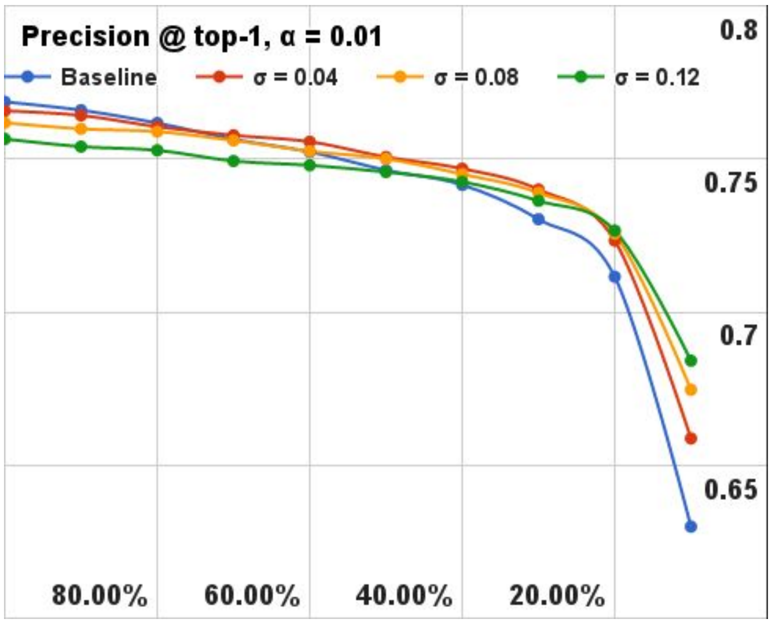
Big gain for large-scale image retrieval systems → ST deployed in Google Image Search

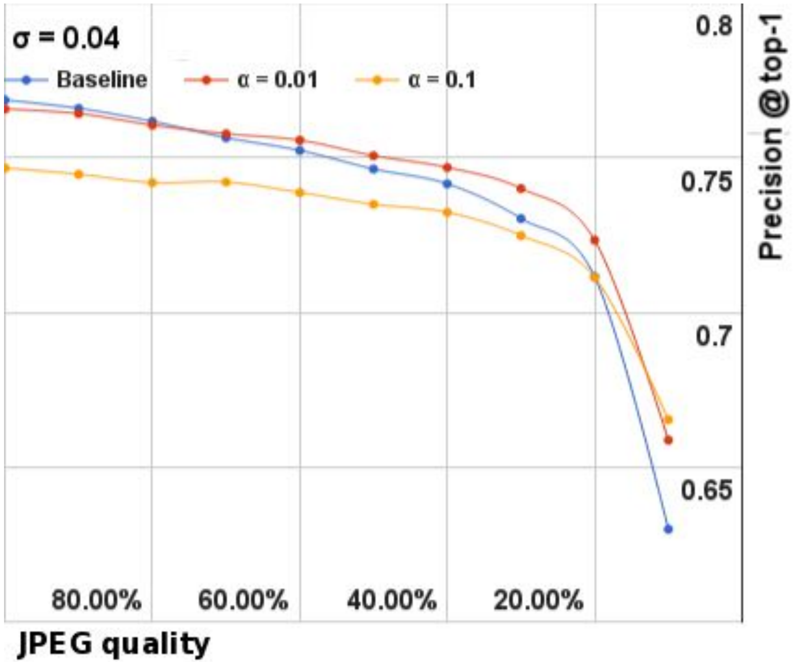




Improving Robustness via Stability Training

Stability training improves **classification** performance by 2-3% on corrupted images





Detecting Adversarial Examples via Neural Fingerprinting

S. Dathathri, SZ*, R. Murray, Y. Yue*

Detecting adversarial examples

Can we **detect** crafted perturbations $x' = x + dx$ that fool our model $f(x)$?

$f\Bigg($  $\Bigg) = \begin{pmatrix} 0.01 \\ \vdots \\ 0.15 \\ \vdots \\ 0.02 \end{pmatrix}$

non-adversarial :)

fawn

Detecting adversarial examples

Can we **detect** crafted perturbations $x' = x + dx$ that fool our model $f(x)$?

$f\Bigg($  $\Bigg) = \begin{pmatrix} 0.03 \\ \vdots \\ 0.08 \\ \vdots \\ 0.01 \end{pmatrix}$

adversarial example!

~~monkey~~

Adversarial examples

Given the data (x, y) loss function L and model parameters θ, an **attacker** tries to find $x' = x + dx$ such that:

$$\max_{x' : ||x - x'||_2 < \delta} L(x', f(x'), y^*; \theta)$$

A **defender** tries to find a θ, mechanism, …, to ensure no solutions x' exist within distance δ of x.

2014 - ... : ongoing "arms race".

Many attacks and defenses have been proposed in recent years.

Many defenses have been broken by stronger attacks.

Hard to (theoretically) guarantee robustness.

Related Work: Attacks

- Fast-gradient-sign method [Goodfellow et al, 2014]

$$x' = x + \epsilon \cdot \mathrm{sign}\frac{dL(x,y;\theta)}{dx}$$

- Basic Iterative Method [Kurakin et al, 2016]
- Projected Gradient Descent [Madry et al, 2017]
- Jacobian Saliency Map [Papernot et al, 2015]
- Carlini-Wagner L_2 [Carlini, Wagner, 2016]

$$\min_{dx} ||dx||_p + c \cdot f(x+dx)$$

$$\text{such that } x + dx \in [0,1]^n$$

$$f \text{ chosen such that } f(x+dx) \leq 0 \Leftrightarrow C(x+dx) = t$$

- SPSA (gradient-free)

Related Work: Defenses

Robust prediction

- Convex relaxations to maximize robustness, formally certify robustness for small perturbations, Raghunathan et al. (2018); Kolter & Wong (2017).
- Randomization (Xie et al., 2018)
- Non-differentiable nonlinearity (Buckman et al., 2018)
- Generative Adversarial Networks for denoising images (Song et al., 2018; Pouya Samangouei, 2018)

Robust detection

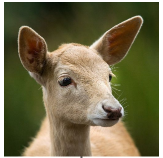
- Auxiliary classifier based on local intrinsic dimensionality (LID), (Ma et al., 2018)
- Kernel Density (KD), Bayesian-Uncertainty (BU) (Feinman et al., 2017)

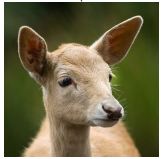
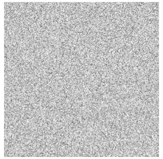
Most have been broken, e.g., using stronger attackers (Carlini & Wagner, 2017a; Athalye et al., 2018)

Neural Fingerprinting

Assume we've chosen some *fingerprint* Δx, Δy, and a trained model f.

NFP does "local consistency check": check if model behaves "as expected" around input x.

x 

$x + \Delta x$  $+$ 

$$f(x) = \begin{pmatrix} 0.01 \\ \vdots \\ 0.15 \\ \vdots \\ 0.02 \end{pmatrix}$$

$$f(x + \Delta x) = \begin{pmatrix} 0.03 \\ \vdots \\ 0.08 \\ \vdots \\ 0.01 \end{pmatrix}$$

$\approx \Delta y^j$ for some j?

yes → x is ok :)

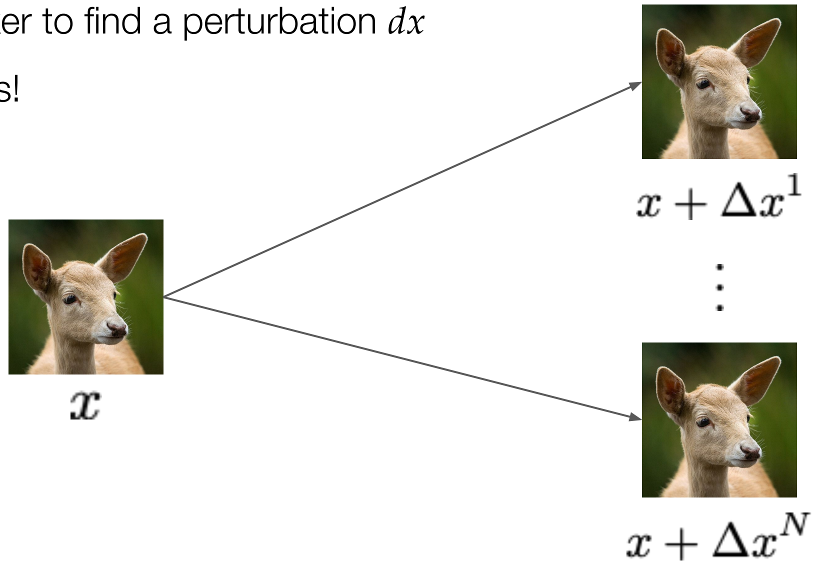
no → x is adversarial!

Neural Fingerprinting

"Local consistency check" using N (secret) fingerprints (prediction on J classes).

Intuition: it becomes increasingly hard for an attacker to find a perturbation dx that conforms with a collection of (secret) fingerprints!

$$\chi^{i,j} = (\Delta x^i, \Delta y^{i,j})$$
$$i = 1 \ldots N, j = 1 \ldots J$$



$$?\exists j : \frac{1}{N} \sum_{i=1}^{N} ||F(x, \Delta x) - \Delta y^{i,j}||_2^2 < \tau$$
$$F(x, \Delta x) = f(x + \Delta x) - f(x)$$

Training with NFP

Train network to behave according to fingerprints on real examples

$$\min_{\theta} \sum_{(x,y)} \left(L_0(x, y; \theta) + \alpha L_{fp}(x, y, \chi; \theta) \right)$$

$$L_{fp}(x, y, \chi; \theta) = \sum_{i=1}^{N} ||F(x, \Delta x^i) - \Delta y^{i,k}||_2^2$$

$$F(x, \Delta x) = f(x + \Delta x) - f(x)$$

Here k is the ground truth class for example (x, y).

Detection with NFP

In practice, using the (normalized) logits $h(x)$ is convenient.

Algorithm 1 *NeuralFP*

1: **Input**: example x, comparison function D (see Eqn 9).
2: **Input**: threshold $\tau > 0$.
3: **Input**: (secret) $\left\{\left(\Delta x^i, \Delta y^{i,j}\right)\right\}_{i=1\ldots N, j=1\ldots K}$.
4: **Output**: `accept` / `reject`.
5: **if** $\exists j : D(x, f, \xi^{i,j}) \leq \tau$ **then**
6:     **Return:** `accept` # *x is real*
7: **else**
8:     **Return**: `reject` # *x is fake*
9: **end if**

$$D(x, f, \xi^{\cdot,j}) = \frac{1}{N} \sum_{i=1}^{N} \|F(x, \Delta x^i) - \Delta y^{i,j}\|_2$$

$$F(x, \Delta x^i) = \varphi(x + \Delta x^i) - \varphi(x),$$

$$\varphi(x) = \frac{h(x)}{\|h(x)\|},$$

Choosing fingerprints

Choose Δx, Δy by random sampling.

$$\Delta x^i \sim \mathbb{N}(0, \sigma^2)$$
$$\Delta y^{i,k}_{l \neq k} = -\alpha(2p - 1)$$
$$\Delta y^{i,k}_{l = k} = \beta(2p - 1)$$
$$p \sim \mathrm{Bern}\left(\frac{1}{2}\right)$$

p resampled for each i.

$\alpha = 0.25$, $\beta = 0.75$ yields good results, but method is robust to this choice.

Random sampling means as little is assumed → minimal information to attacker.

Attacker might or might not know the chosen fingerprints.

Linear Models Guarantees

SVM: 4 fingerprints (dx, dy) maximize true positive region (blue).

NFP detects whether the predicted class is correct (= choice of sign).

Detect adversarial examples too close / far from the decision boundary.

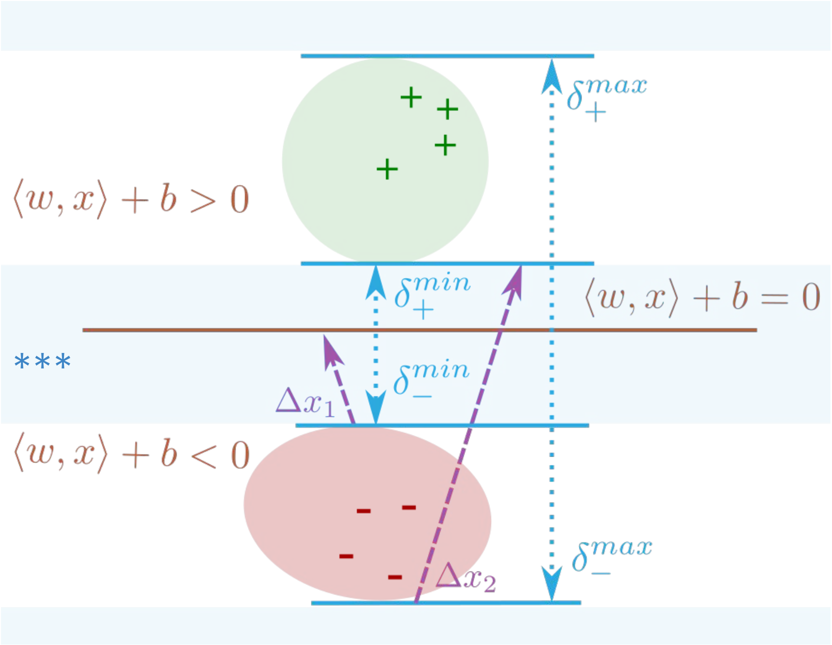
Example:

For Δx_1 : sign $f(x)$ = sign $f(x + \Delta x_1)$.

Hence, Δx_1 excludes all adversarials in ***,

because sign $f(x + \Delta x_1)$ is always on the other

side of the decision boundary.



Fingerprint Loss for Nonlinear Models

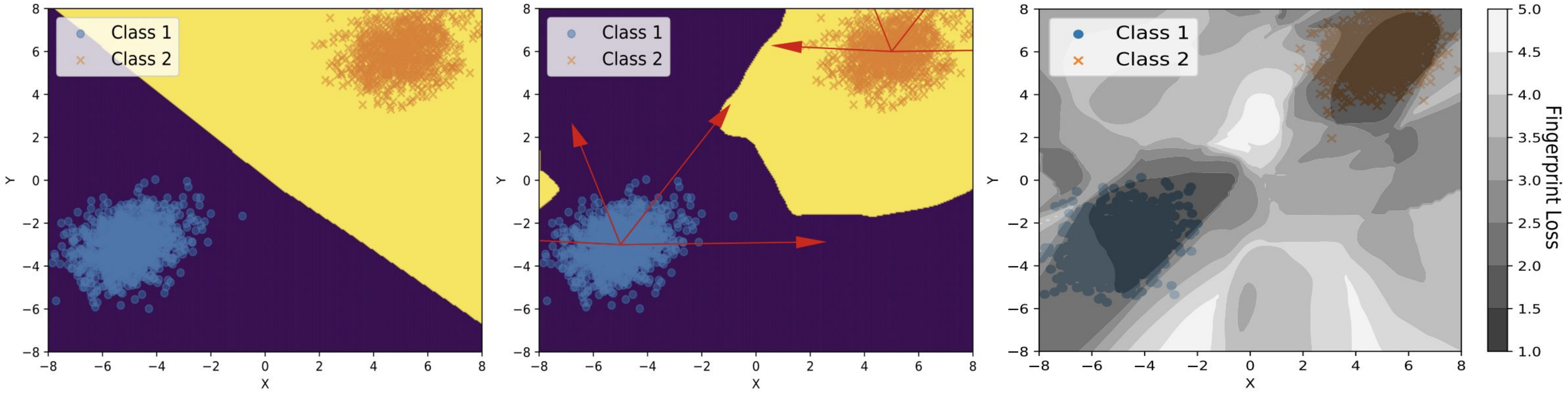
Train a neural net on binary classification to also fit **fingerprints**.

NFP introduces tight fingerprint loss landscape around data.

Hard for attackers to find areas with low fingerprint loss.

$$L_{fp}(x, y, \chi; \theta) = \sum_{i=1}^{N} ||F(x, \Delta x^i) - \Delta y^{i,k}||_2^2$$

$$F(x, \Delta x) = f(x + \Delta x) - f(x)$$



Without NFP

With NFP

Security Scenarios

NFP is a **blackbox defense**: it does not assume any knowledge of the attacker.

Furthermore, we can evaluate NFP in various **attack scenarios:**

Attacker knows θ	Attacker knows NFP	**Attack** scenario
No	No	Blackbox
Yes	No	Greybox** / partial-whitebox
Yes	Yes	(Adaptive) whitebox

For instance, in the adaptive whitebox scenario, an attacker tries to solve

$$\min_{x'} ||x - x'||_2 + \gamma_1 L_{CW}(x') - \gamma_2 L_{fp}(x', y^*, \chi; \theta), \quad \gamma_1, \gamma_2 > 0$$

Here, L_{CW} is a loss function that is a proxy for misclassification of x'.

Experimental Setup

1. Sample fingerprints.
2. Train model to high task accuracy + low fingerprint loss.
3. On test-set, generate adversarial examples for images that are correctly classified.
4. Check how many real/adversarial images are correctly classified by NFP.

Sanity Checks

[Carlini, ICML 2018] suggested several tests to check detection algorithms.

1. **Trivial attacks do not work:** random sampling around data does not fool NFP.
2. **Good attacks exist:** test-images fool NFP.
3. **A good unbounded attack should reduce accuracy to 0%.**

   In practice, this depends on the strength of known attacks.

   Unpublished attacks do start to break detection (accuracy < 50%), **but** are computationally expensive.

   - PGD with 30k steps + scheduled gamma-decay + tweaks
   - 70% AUC-ROC at $\delta = 0.25$
   - < 50% AUC-ROC at $\delta = 1$
   - Ongoing experiments!

Greybox: Near-Perfect Detection of SotA attacks

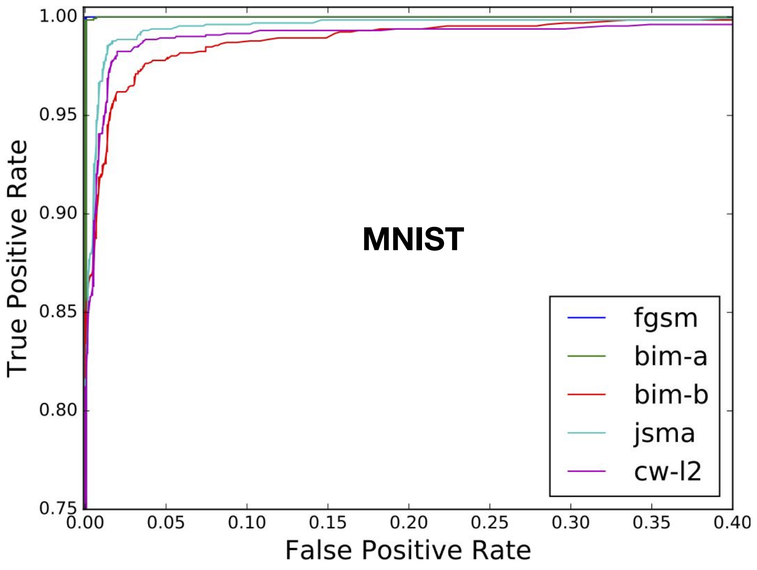
Near-perfect AUC-ROC on SotA attacks, for MNIST, CIFAR, MiniImagenet-20.

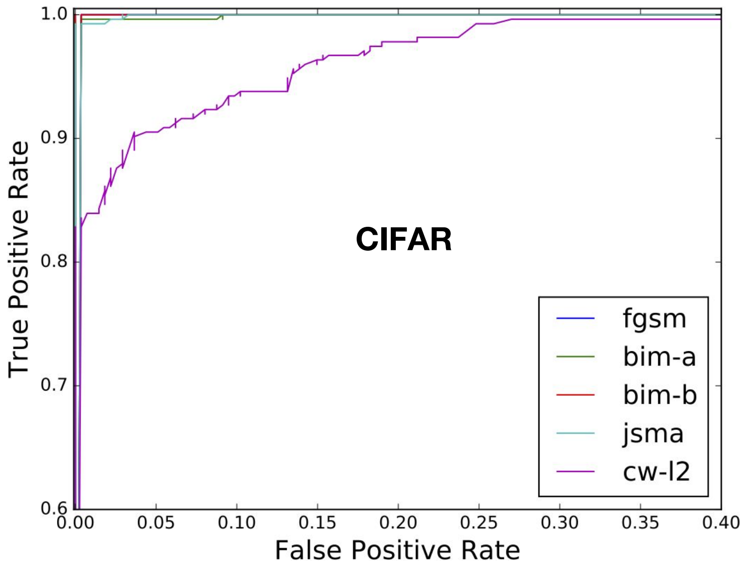
Data	Method	FGSM	JSMA	BIM-a	BIM-b	CW-L_2
MNIST	LID	99.68	96.36	99.05	99.72	98.66
	KD	81.84	66.69	99.39	99.84	96.94
	BU	27.21	12.27	6.55	23.30	19.09
	KD+BU	82.93	47.33	95.98	99.82	85.68
	NeuralFP	**100.0**	**99.97**	**99.94**	**99.98**	**99.74**
CIFAR-10	LID	82.38	89.93	82.51	91.61	93.32
	KD	62.76	84.54	69.08	89.66	90.77
	BU	71.73	84.95	82.23	3.26	89.89
	KD+BU	71.40	84.49	82.07	1.1	89.30
	NeuralFP	**99.96**	**99.91**	**99.91**	**99.95**	**98.87**

Data	FGSM	BIM-b
MiniImagenet-20	99.96	99.68

Greybox: Near-Perfect Detection of SotA attacks

Near-perfect AUC-ROC on SotA attacks, for MNIST, CIFAR, MiniImagenet-20.





Adaptive Whitebox

Near-perfect AUC-ROC on all state-of-the-art attacks.

Existing detection methods (KD, BU, LID) are too slow, and are effectively broken (< 10% AUC-ROC).

Executing attacks for large adversarial perturbation bounds δ becomes computationally expensive.

Attacks use a binary search over attack-parameters and exhaustive # steps.

Data	Method	Adaptive-FGSM	Adaptive-BIM-b	Adaptive-CW-L_2	Adaptive-CW-L_2 ($\gamma_2 = 1$)	Adaptive-SPSA
MNIST	*NeuralFP*	99.91	99.37	95.04	99.17	99.94
CIFAR-10	*NeuralFP*	99.99	99.92	97.19	97.56	99.99

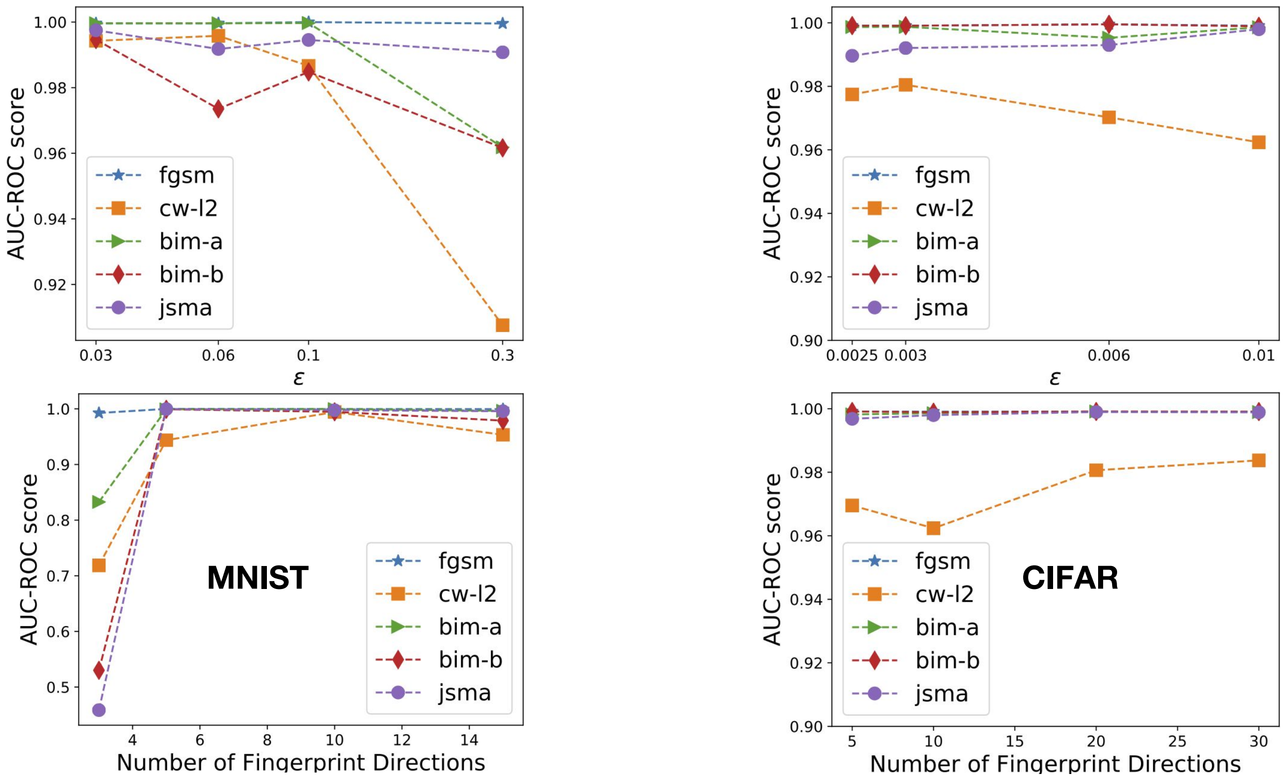
AUC-ROC scores

MNIST: $(\varepsilon, N) = (0.1, 10)$ vs FGSM, SPSA. $(\varepsilon, N) = (0.05, 20)$ vs BIM-b, CW-L2.

CIFAR-10: $(\varepsilon, N) = (0.003, 30)$.

Broad Robust Performance

AUC-ROC performance across wide range of hyperparameters



State-of-the-art attacks fail at breaking NFP

Even with extreme computation budgets, SotA attacks still can’t fool NFP.

Attack	Distortion bound	Iteration steps	Bisection steps	Step-size	AUC-ROC (%)
CarliniL2-FP (N=30, vareps=0.003)	Unbounded	20000	20, 10		**94.12**
CarliniL2-FP (N=30, vareps=0.003)	Unbounded	20000	15, 9		**95.48**
CarliniL2-FP (N=50, vareps=0.003)	Unbounded	20000	20, 10		**95.56**
Adaptive-PGD (l-2)	10	Steps = 100 Restarts = 5	6	0.5	**99.37**
Adaptive-PGD (l-inf)	0.25	Steps = 1000 Restarts = 1	6	0.005	**99.71**
Adaptive-PGD (l-inf)	1.0	Steps = 150 Restarts = 5	6	0.01	**99.74**
Adaptive-SPSA	0.05	Steps = 1000	20	0.01, delta = 0.01	**99.84**

Contributions

Neural Fingerprinting is a very promising basis for defending neural networks.

Easy to implement and execute.

Gives state-of-the-art detection of adversarial examples.

Works very well in greybox / whitebox settings.

Current state-of-the-art attacks are not strong enough.

Passes all sanity checks for detection mechanisms (with tweaked attacks).

Future work

Extend defenses to further improve detection rates and robustness against stronger attacks.

Develop stronger attacks.

Give theoretical guarantees: stop the arms-race.

- for bounded attacks
- for nonlinear models

Characterizing the loss-function geometry of neural networks.

Acknowledgements




Yisong Yue
www.yisongyue.com

Sumanth Dathathri

Yang Song

Ian Goodfellow

Caltech

Google