Robust Neural Networks

Dirk Nowotka

August 2019

Kiel University, Germany

Evaluating Robust Neural Networks

Dirk Nowotka

August 2019

Kiel University, Germany

What are we talking about?

What is a good defence mechanism?

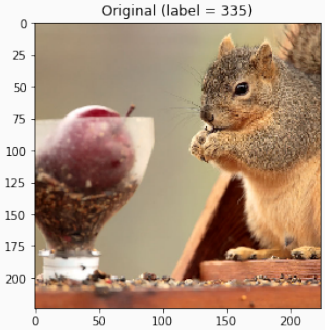
How should we evaluate a defence mechanism?

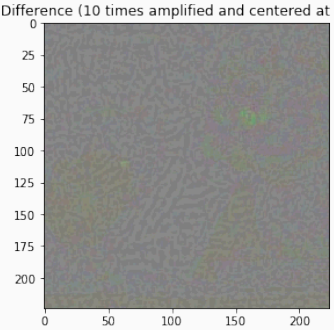
How about an example?

What is the take away message?

What are we talking about?

Adversarial Example





Setting

Object Domain $\mathcal{D} \subseteq \mathbb{R}^q$

Similarity Measure $s\colon \mathcal{D} \times \mathcal{D} \to \mathbb{R}_{\geq 0}$, often a norm like the Euclidean (ℓ_2) or Chebyshev (ℓ_∞) norm, or just counting non-zero entries (ℓ_0)

Neural Network here, focus on classification $c\colon \mathcal{D} \to \{1, \dots j\}$ with j classes

Test Sequence $S = (\mathcal{D} \times \{1, \dots j\})^*$

Adversarial Example given $x \in \mathcal{D}$ correctly classified, then x' is adversarial if $c(x) \neq c(x')$ and $s(x, x') < \epsilon$

Defense Mechanism function d to modify a classifier c to $d(c)$

What is a good defence mechanism?

Application Settings

In a smart and connected world . . .

- ML is applied in safety critical and security sensitive areas.
- Users rely on models to work as intended (most of the time).
- Models are rolled out on a large scale. They are ubiquitous.
- Applications are often resource conscious (mobile / embedded devices).
- Privacy is an issue. (not considered here)

Properties of a Good Defence Mechanism

Efficacy Ratio of adversarial examples classified correctly over the total amount of adversarial examples applied

Quality Measures accuracy on benign input

Robustness Ratio of a single adversarial example being misclassified over the number of installations of $d(c)$

Efficiency Overhead of running time / space of $d(c)$ compared to c

How should we evaluate a defence mechanism?

What do we need to start with?

- Giving strong evidence to adversarial robustness is difficult.
- Defence mechanisms are derived from certain attack mechanisms or follow general considerations.
- Evaluating a defence mechanism requires scrutiny, scepticism, and careful analysis.
- Defence can only be evaluated in relation to a threat model.
- IT security research is a good example to learn from.

What do we need to start with?

- Giving strong evidence to adversarial robustness is difficult.
- Defence mechanisms are derived from certain attack mechanisms or follow general considerations.
- Evaluating a defence mechanism requires scrutiny, scepticism, and careful analysis.
- Defence can only be evaluated in relation to a threat model.
- IT security research is a good example to learn from.

A **methodology** needs to be established and commonly agreed on!

Suggested literature:

Papernot et. el., Towards the Science of Security and Privacy in Machine Learning, 2016.

Carlini et. al., On Evaluating Adversarial Robustness, 2019.

Main Principle

- Validation of a scientific theory requires verification or falsification.
- A good defence model is a universal claim.
- Following Karl Popper, genuine scientific claims are (universal and, at least in principle,) falsifiable.

Main Principle

- Validation of a scientific theory requires verification or falsification.
- A good defence model is a universal claim.
- Following Karl Popper, genuine scientific claims are (universal and, at least in principle,) falsifiable.

A defence mechanism therefore cannot exist without a formulation of **context**.

[Karl Popper, Logik der Forschung, 1934.]

Context: Threat Model

Context is formulated as a threat model.

Adversary Goals Against which kind of manipulation do we want to defend against?

Adversary Capabilities Which parts of our system are assumed to be exposed to an attacker?

Adversary Knowledge Which information may the attacker have?

Threat Model: Adversary Goals

Here, an adversary attempts to provoke a misclassification.

Threat Model: Adversary Goals

Here, an adversary attempts to provoke a misclassification.

Let $x, x' \in \mathcal{D}$ and $c(x) = t$ and $c(x') = t'$ with $t \neq t'$ and $s(x, x') < \epsilon$.

Input x' may be such that label t' is

arbitrary: just misclassification, e.g. confuse control of autonomous vehicle.

function of t: source-target-misclassification, e.g. disguise attack on credit rating system.

fixed: indiscrimination, e.g. attack on access control.

Threat Model: Adversary Capabilities

An unconstraint attacker can do arbitrary harm. Restrictive assumptions on the capabilities of an attacker are necessary for a meaningful defence mechanism.

Most common: Attacker has control over input and may apply "small" changes to it.

How do we determine what a small change, $s(x, x') < \epsilon$, is?

Threat Model: Adversary Capabilities

An unconstraint attacker can do arbitrary harm. Restrictive assumptions on the capabilities of an attacker are necessary for a meaningful defence mechanism.

Most common: Attacker has control over input and may apply "small" changes to it.

How do we determine what a small change, $s(x, x') < \epsilon$, is?

Examples for $\mathcal{D}$ representing images:

- ℓ_0 number of pixels changed.
- ℓ_2 square root of the sum of all squared pixel changes.
- ℓ_∞ maximum difference of pixel change.

Certain applications may have very different restrictions, e.g. malware detection.

Threat Model: Adversary Capabilities

Calculating the adversarial risk like

- worst case loss for a given perturbation budget
- average minimum adversarial perturbation

is usually hard.

$$\mathbb{E}_{(x,t)\sim\mathcal{X}}\left(\min_{c(x')\neq t} s(x, x')\right)$$

The average case is easier to calculate by sampling.

Threat Model: Adversary Knowledge

Assumptions on the attacker's knowledge

white-box complete knowledge of model

black-box no knowledge of model, but usually varying degrees of black-box access (queries to model, predicted probabilities, training data, etc.)

Threat Model: Adversary Knowledge

Assumptions on the attacker's knowledge

white-box complete knowledge of model

black-box no knowledge of model, but usually varying degrees of black-box access (queries to model, predicted probabilities, training data, etc.)

Analogously to Kerckhoffs' principle in cryptology:

Complete knowledge of the defence mechanism is assumed.

[Auguste Kerckhoffs, La cryptographie militaire, 1883.]

Further Aspects of a Sound Evaluation Methodology

- Consider only attacks compatible with threat model.
- Be sceptical of your result. (blind spots)
- Try to find an attack which breaks your own defence. (psychologically hard)
  - Assume adaptive adversaries.
  - Test state-of-the-art attacks.
  - Try to come up with a new attack which might break your defence mechanism.
- Share code, models, and data. (reproducible research)

How about an example?

Randomly Perturbed Ensemble Neural Networks (RPENN)

A randomised defence mechanism

- Take m copies of a trained classifier c
- Randomly perturb weights of each copy
- Compose those modified copies into an ensemble via majority voting

Threat Model: Adversary Goals

Untargeted Attack:

Let $x \in \mathcal{D}$ with $c(x) = t$.

The attacker attempts to find an $x' \in \mathcal{D}$ such that $c(x') \neq t$ and $s(x, x') < \epsilon$.

Note: No particular optimisation to minimise $s(x, x')$ was made.

Threat Model: Adversary Capabilities

The attacker may

- manipulate the input of the system.
- use ℓ_0, ℓ_2, and ℓ_∞ similarity measures.

Threat Model: Adversary Capabilities

The attacker may

- manipulate the input of the system.
- use ℓ_0, ℓ_2, and ℓ_∞ similarity measures.

The attacker may not

- interfere with the training process.
- manipulate the application of c or its output.
- interfere with the processing of the output of c.

Threat Model: Adversary Knowledge

The attacker is assumed to have complete knowledge of the

- model (white-box).
- defence mechanism.

Threat Model: Adversary Knowledge

The attacker is assumed to have complete knowledge of the

- model (white-box).
- defence mechanism.

The attacker is assumed to have no knowledge of the

- training data.
- seeds used to implement randomisation.

Further Aspects: Adaptive Adversaries

Six different attack methods were applied

- Fast Gradient Sign Method (FGSM) [Goodfellow, Shlens, Szegedy 2014]
- Basic Iterative Method, ∞-norm (BIM$_{\ell_\infty}$) [Kurakin, Goodfellow, Bengio 2016]
- DeepFool Attack (DFA) [Moosavi-Dezfooli, Fawzi, Frossard 2016]
- Saliency Map Attack (SMA) [Papernot et. al. 2016]
- L-BFGS Attack (L-BFSG) [Szegedy et. al. 2013]
- Carlini Wagner Attack, 2-norm (CW$_{\ell_2}$) [Carlini, Wagner 2017]

Randomly Perturbed Ensemble Neural Networks (RPENN)

A randomised defence mechanism

- Take m copies of a trained classifier c
- Randomly perturb weights of each copy
- Compose those modified copies into an ensemble via majority voting

Randomly Perturbed Ensemble Neural Networks (RPENN)

Let c be a trained classifier with weights $w_1, w_2, \ldots w_k$. Classify input x as follows

1. Choose new weights independently at random

$$\begin{array}{ccc} w_1^1 \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^1 \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right) \\ \vdots & \ddots & \vdots \\ w_1^m \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^m \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right), \end{array}$$

   where $\lambda \in \mathbb{R}_{>0}$ is a user defined parameter and $\mathcal{N}\left(w_i, (\lambda \cdot w_i)^2\right)$ is a normal distribution with mean w_i and variance $(\lambda \cdot w_i)^2$.

Randomly Perturbed Ensemble Neural Networks (RPENN)

Let c be a trained classifier with weights $w_1, w_2, \ldots w_k$. Classify input x as follows

1. Choose new weights independently at random

$$\begin{array}{ccc} w_1^1 \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^1 \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right) \\ \vdots & \ddots & \vdots \\ w_1^m \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^m \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right), \end{array}$$

   where $\lambda \in \mathbb{R}_{>0}$ is a user defined parameter and $\mathcal{N}\left(w_i, (\lambda \cdot w_i)^2\right)$ is a normal distribution with mean w_i and variance $(\lambda \cdot w_i)^2$.
2. Generate m variations $c_1, c_2, \ldots c_m$ of c where c_i has weights $w_1^i, w_2^i, \ldots w_k^i$.

Randomly Perturbed Ensemble Neural Networks (RPENN)

Let c be a trained classifier with weights $w_1, w_2, \ldots w_k$. Classify input x as follows

1. Choose new weights independently at random

$$\begin{array}{ccc} w_1^1 \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^1 \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right) \\ \vdots & \ddots & \vdots \\ w_1^m \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^m \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right), \end{array}$$

   where $\lambda \in \mathbb{R}_{>0}$ is a user defined parameter and $\mathcal{N}\left(w_i, (\lambda \cdot w_i)^2\right)$ is a normal distribution with mean w_i and variance $(\lambda \cdot w_i)^2$.
2. Generate m variations $c_1, c_2, \ldots c_m$ of c where c_i has weights $w_1^i, w_2^i, \ldots w_k^i$.
3. Compute $c_1(x), c_2(x), \ldots c_m(x)$.

Randomly Perturbed Ensemble Neural Networks (RPENN)

Let c be a trained classifier with weights $w_1, w_2, \ldots w_k$. Classify input x as follows

1. Choose new weights independently at random

$$\begin{array}{ccc} w_1^1 \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^1 \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right) \\ \vdots & \ddots & \vdots \\ w_1^m \sim \mathcal{N}\left(w_1, (\lambda \cdot w_1)^2\right) & \ldots & w_k^m \sim \mathcal{N}\left(w_k, (\lambda \cdot w_k)^2\right), \end{array}$$

   where $\lambda \in \mathbb{R}_{>0}$ is a user defined parameter and $\mathcal{N}\left(w_i, (\lambda \cdot w_i)^2\right)$ is a normal distribution with mean w_i and variance $(\lambda \cdot w_i)^2$.
2. Generate m variations $c_1, c_2, \ldots c_m$ of c where c_i has weights $w_1^i, w_2^i, \ldots w_k^i$.
3. Compute $c_1(x), c_2(x), \ldots c_m(x)$.
4. Combine $c_1(x), \ldots c_m(x)$ to a single output t, e.g. by averaging or majority vote.
5. Return t.

Remarks

RPENN parameters λ (relative deviation) and m (ensemble size)

Remarks

RPENN parameters λ (relative deviation) and m (ensemble size)

The case $m = 1$ is similar to L^* by Gu and Rigazio without perturbating the input.

However, RPENN uses **weight-individual Gaussian distributions**, that is, the added noise is relative to the size of each weight. Gu and Rigazio use one Gaussian distribution for all weights influencing small weights more than large weights.

[Gu, Rigazio, Towards deep neural network architectures robust to adversarial examples, 2014]

Properties the Defence Mechanism

Efficacy Ratio of adversarial examples classified correctly over the total amount of adversarial examples applied

Quality Measures accuracy on benign input

Robustness Ratio of a single adversarial example being misclassified over the number of installations of $d(c)$

Efficiency Overhead of running time / space of $d(c)$ compared to c

Ratio of correctly classified adversarial input for $d(c)$

Efficacy measure $\mathcal{E}$ for a classifier c, an attacker $\mathcal{A}$, a test sequence S, and a defence mechanism d

$$\mathcal{E}_{c,\mathcal{A},S}(d) = \frac{\mathbb{M}(d(c)(\mathcal{A}_c(S)), S)}{|\mathcal{A}_c(S)|}$$

where

$\mathcal{A}_c(S)$ sequence of adversarial examples which $\mathcal{A}$ derives from S for c

$\mathbb{M}(L, S)$ number of matching labels of label sequence L and S

Ratio of correctly classified test input for $d(c)$

Quality measure $\mathcal{Q}$ for a classifier c, a test sequence S, and a defence mechanism d

$$\mathcal{Q}_{c,S}(d) = \frac{\mathbb{M}(d(c)(S), S)}{|S|}$$

where

$\mathbb{M}(L, S)$ number of matching labels of label sequence L and S

Efficiency

- No precise estimate here.
- RPENN adds an overhead linear in the network and the ensemble size.
- We saw no significant performance penalty during our tests for small and medium ensemble sizes.
- Problematic for resource conscious applications, like mobile or embedded devices.

Setup

Attempt of realistic and relevant results.

Data set ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC 2012)

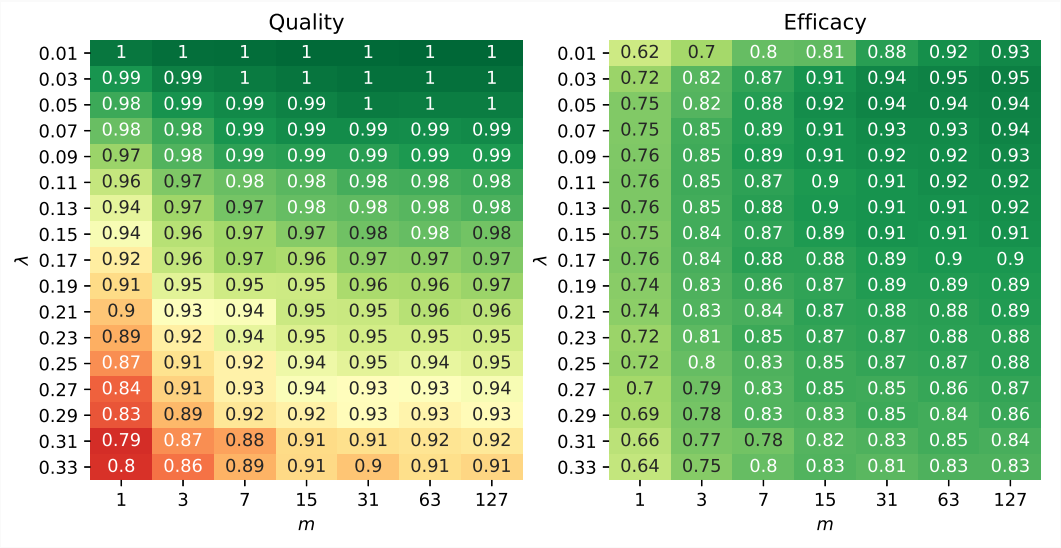
- 50000 color images
- 224×224 resolution
- compare to 28×28 gray scale images of MNIST data set

Model VGG19 convolutional network

ML framework Keras

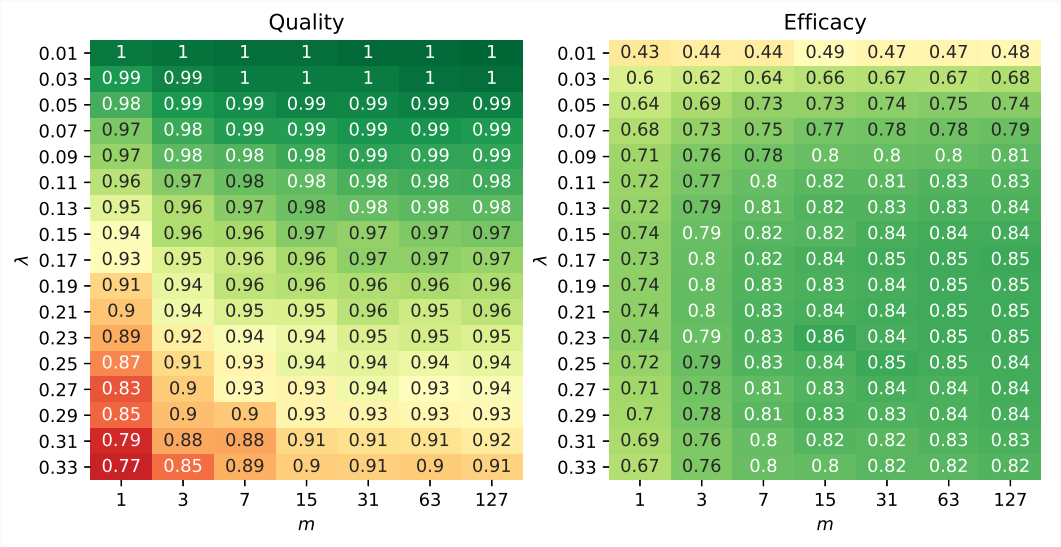
AE generator FoolBox

Results: Basic Iterative Method ∞-norm

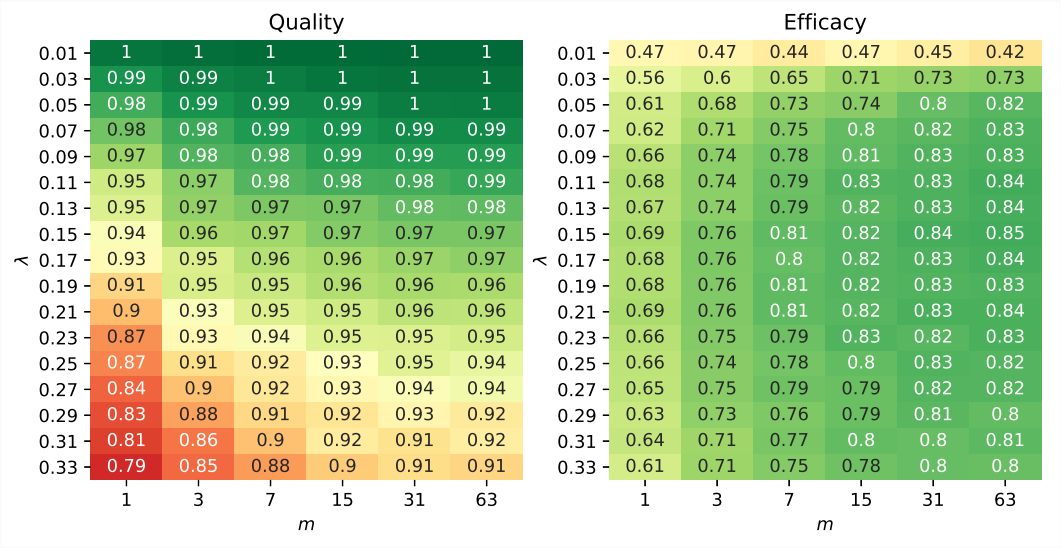


Results: DeepFool Attack

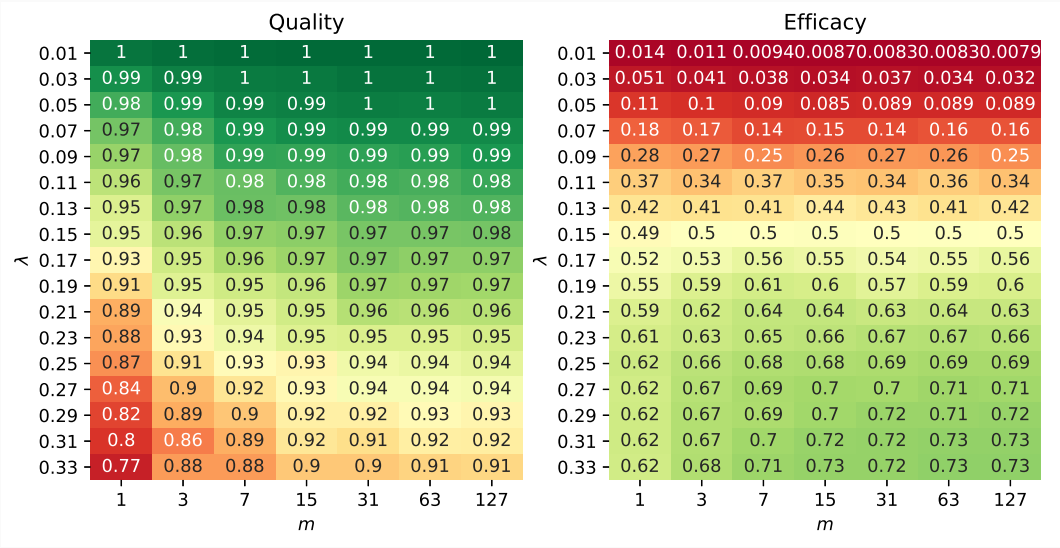
Results: Carlini Wagner Attack, 2-norm



Results: Saliency Map Attack



Results: L-BFGS Attack



Results: Fast Gradient Sign Method

Portion of adversarial examples misclassified by at least $\lfloor q \cdot n \rfloor$ installations

Robustness measure $\mathcal{R}$ for a classifier c, an attacker $\mathcal{A}$, a test sequence S, a defence mechanism d, a robustness level $0 < q \leq 1$, and n installations

$$\mathcal{R}_{c,\mathcal{A},S}(d, q, n) = \frac{|\{A_i \mid \mathbb{M}(\langle L_{i,1}, \ldots L_{i,n} \rangle, \langle A_i, \ldots A_i \rangle) \geq \lfloor q \cdot n \rfloor\}|}{|\mathcal{A}_c(S)|}$$

where

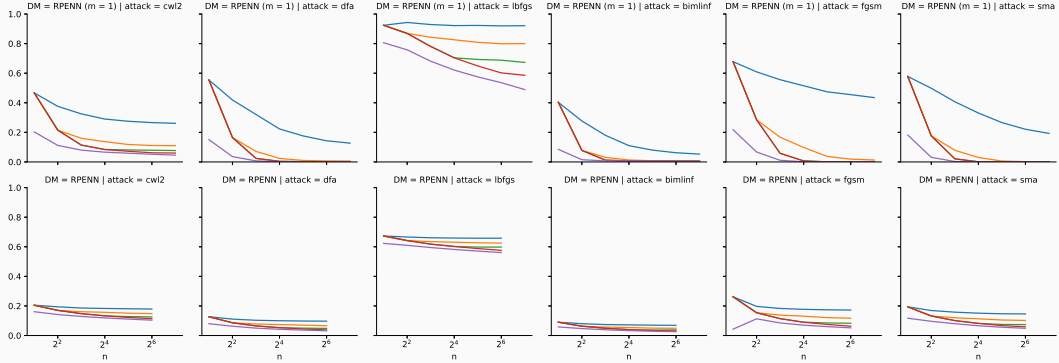
A shorthand for $\mathcal{A}_c(S)$

L defined as $\langle d_{\sigma_1}(c)(A)^T, \ldots d_{\sigma_n}(c)(A)^T \rangle$

$d_{\sigma_1}, \ldots d_{\sigma_n}$ sequence of installations of a randomised defence mechanism d

$\sigma_1, \ldots \sigma_n$ sequence of random seeds

Results: Robustness



$\mathcal{R}(d, q, n)$, for $q \in \{0.50, 0.80, 0.95, 0.99, 1.00\}$ and d such that $m = 1$, $\lambda = 0.03$ (top) and $m = 7$ and $\lambda = 0.05$ (bottom). Quality $\mathcal{Q} > 0.99$ and attack methods CW_{ℓ_2} DFA, L-BFGS and BIM_{ℓ_∞}, FGSM, and SMA.

Related Defence Mechanisms

Stochastic Feedforward by Jin, Dundar, Culurciello 2015

- injection of noise in input and hidden layers during training

Random Self-Ensemble by Liu, Cheng, Zhang, Hsieh 2017

- inject noise in activation, not weights

L1, L* by Gu and Rigazio 2014

- not clear where noise is added to layers, no implementation available
- random samples taken from one Gaussian distribution (independent of weights)

Considerations

Done

- Proposed new defence mechanism **RPENN**
- Evaluation guided by common methodology
- Extended common methodology by **criteria for defence mechanisms**
- Performed **large parameter search**

To be done

- Further evaluation in **different scenarios**, however ILSVRC2012 is quite comprehensive
- Evaluation against **non-gradient based attacks**
- Consider criticism of gradient based attacks [Athalye, Carlini, Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples, 2018]

What is the take away message?

Conclusion

Do we have a truly robust neural network?

Conclusion

Do we have a truly robust neural network?

Not really.

Conclusion

Do we have a truly robust neural network?

Not really.

But, we made progress on recognising one when we see it.

Conclusion

Do we have a truly robust neural network?

Not really.

But, we made progress on recognising one when we see it.

— End of Talk —