Artificial Intelligence III: Advanced Deep Learning Methods

Ana Neacșu & Vlad Vasilescu

Chapter 2: Defense strategies

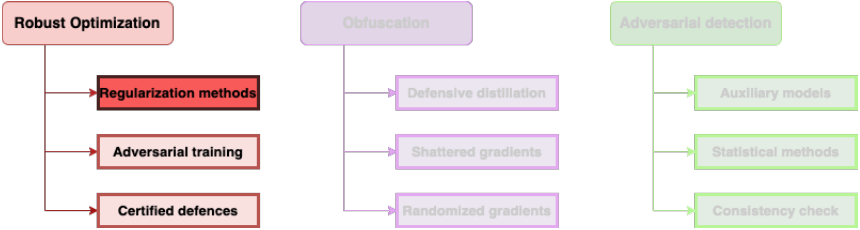
National University of Science and Technology POLITEHNICA Bucharest, Romania
BIOSINF Master Program

October 2025

Introduction

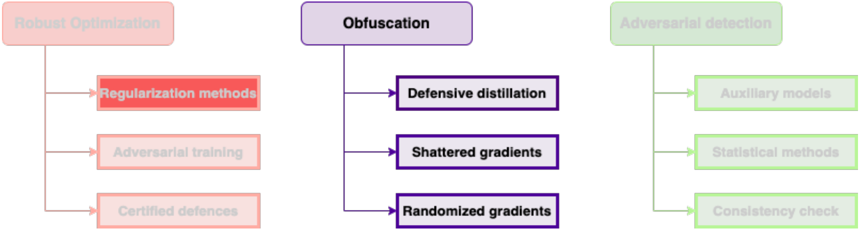
Defense strategies

Three main methodologies for creating models robust to adversarial perturbations.



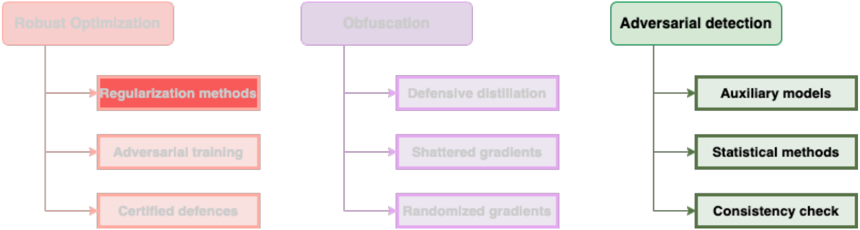
Defense strategies

Three main methodologies for creating models robust to adversarial perturbations.



Defense strategies

Three main methodologies for creating models robust to adversarial perturbations.



Robustness Quantification

Consider an attacker $\mathtt{A}$ acting on input pair (x, y) and producing adversarial sample x' through a model parametrized by θ:

$$\mathtt{A} : (x, y) \xrightarrow{\theta} x'$$

Attack-dependent Measures:

- Attack Success Rate (ASR):

$$\mathtt{ASR} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{test}}}\left[\mathbb{1}\{f_\theta(x') \neq y\}\right] \tag{1}$$

- Robust Accuracy (RA):

$$\mathtt{RA} = 1 - \mathtt{ASR} \tag{2}$$

- Average Perturbation:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{test}}}\left[\|x - x'\| \cdot \mathbb{1}\{f_\theta(x') \neq y\}\right] \tag{3}$$

Robustness Quantification

Attack-free Measures

- Certified Robust Accuracy (CRA):

$$\mathtt{CRA}_\epsilon = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{test}}}\left[\mathbb{1}\{f_\theta(x+z) = y, \forall z, \ \|z\|_p \leq \epsilon\}\right] \tag{4}$$

  Measures how many points are robust to **any perturbation** within $\mathbb{B}_p(x, \epsilon)$

- Certified Radius (CR):

$$\mathtt{CR}(x, y) = \sup\{\epsilon \geq 0 \,|\, f_\theta(x+z) = y, \forall z, \ \|z\|_p \leq \epsilon\} \tag{5}$$

  Measures the maximum radius under which f_θ will correctly classify x.

- Average Certified Radius (ACR):

$$\mathtt{ACR} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{test}}}[\mathtt{CR}(x, y) \cdot \mathbb{1}\{f_\theta(x) = y\}] \tag{6}$$

Naive defenses

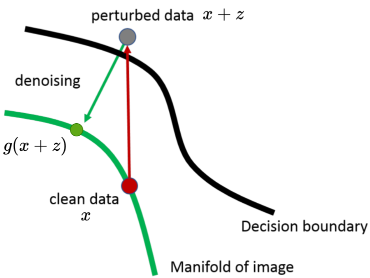
Pre-processing as a defense

Idea : Can some pre-processing techniques be used as a defense?

Consider an adversarial example $x' = x + z$ for which $f_\theta(x+z) \neq f_\theta(x)$. We are interested in finding $g(x) : \mathcal{X} \to \mathcal{X}$ s.t. $(f_\theta \circ g)(x+z) = (f_\theta \circ g)(x) = y$

This can be seen as a:

- Denoising problem
- Geometry projection



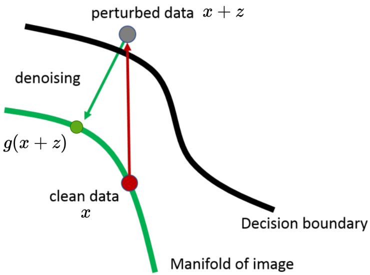
Pre-processing as a defense

Idea : Can some pre-processing techniques be used as a defense?

Consider an adversarial example $x' = x + z$ for which $f_\theta(x+z) \neq f_\theta(x)$. We are interested in finding $g(x) : \mathcal{X} \to \mathcal{X}$ s.t. $(f_\theta \circ g)(x+z) = (f_\theta \circ g)(x) = y$

This can be seen as a:

- Denoising problem
- Geometry projection



Input transform

Countering Adversarial Images using Input Transformations – using *Total Variance* minimization:

Objective

$$\underset{z}{\text{minimize}} \|(1 - X) \odot (z - x)\|_2 + \lambda \text{TV}(z), \tag{7}$$

where

- TV – Total Variation
- $\text{TV}(z) = \|D_{\downarrow}(z)\|_1 + \|D_{\rightarrow}(z)\|_1$
- $\lambda \in (0, 1)$
- $\odot$ is point-wise multiplication

Some examples

- Defense against Adversarial Attacks Using High-Level Representation Guided Denoiser

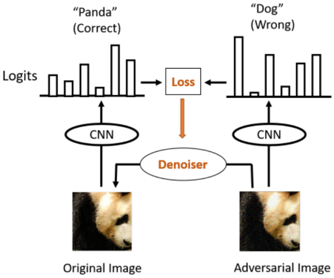
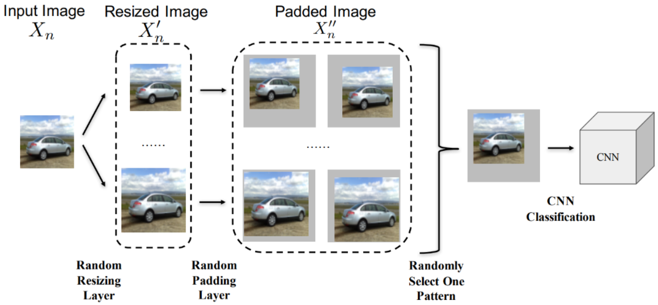


Figure 1: The idea of high-level representation guided denoiser. The difference between the original image and adversarial image is tiny, but the difference is amplified in high-level representation (logits for example) of a CNN. We use the distance over high-level representations to guide the training of an image denoiser to suppress the influence of adversarial perturbation.

Random Transforms

Idea : Since attack is so specific along one direction, we could use a random perturbation to change its path.

Observation: In a high dimensional space, a small change in *direction* can have a huge impact in the *destination*.

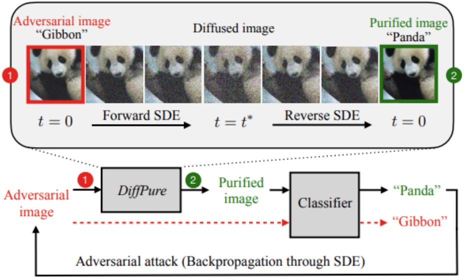


[Source]

Adversarial Purification

Paper: Diffusion Models for Adversarial Purification

Idea : Use pre-trained diffusion models (e.g. *Stable Diffusion*) to reconstruct a noisy version of the adversarial image



[Source]

Problems : high computational overhead + some adversarial corruptions might target the reverse SDE to produce worse purified images

Adversarial Training

Adversarial Training (AT)

Idea : feed the classifier with adversarial examples during training to boost its robustness against perturbations

Paper: Towards Deep Learning Models Resistant to Adversarial Attacks

Formulated as a min-max problem

$$\underset{\theta}{\text{minimize}} \quad \mathbb{E}_{(\boldsymbol{x},\mathbf{y})\sim\mathcal{D}} \left[\max_{\delta\in\Omega} \mathcal{L}(\theta, \boldsymbol{x}+\delta, \mathbf{y}) \right]$$

- $\mathcal{D}$ is the training set
- $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$ are in input and the associated ground truth.
- $\mathbb{E}$ symbolizes expectation over samples
- δ is the adversarial perturbation within $\mathcal{B}(x, \epsilon)$
- θ encompasses network's weights
- $\mathcal{L}$ is a chosen attack objective

AT General Algorithm

1. Generate adversarial examples

At epoch t, for a subset $\mathcal{D}' \subseteq \mathcal{D}$ generate adversarial perturbations $\delta_i^{(t)}$ given the current model parameters $\theta^{(t)}$ and perturbation budget ϵ

2. Augument training set (max. problem)

Define $\mathcal{D}_{adv} = \{(x_i + \delta_i^{(t)}, y_i) | x_i \in \mathcal{D}'\}$ and construct a new training set $\mathcal{D}^{(t)} = \mathcal{D}_{adv} \cup \mathcal{D}$

3. Adversarial training (min. problem)

Train on augumented $\mathcal{D}^{(t)}$ for $N \geq 1$ epochs, then repeat from Step 1.

AT General Algorithm

1. Generate adversarial examples

At epoch t, for a subset $\mathcal{D}' \subseteq \mathcal{D}$ generate adversarial perturbations $\delta_i^{(t)}$ given the current model parameters $\theta^{(t)}$ and perturbation budget ϵ

2. Augument training set (max. problem)

Define $\mathcal{D}_{adv} = \{(x_i + \delta_i^{(t)}, y_i) | x_i \in \mathcal{D}'\}$ and construct a new training set $\mathcal{D}^{(t)} = \mathcal{D}_{adv} \cup \mathcal{D}$

3. Adversarial training (min. problem)

Train on augumented $\mathcal{D}^{(t)}$ for $N \geq 1$ epochs, then repeat from Step 1.

AT General Algorithm

1. Generate adversarial examples

At epoch t, for a subset $\mathcal{D}' \subseteq \mathcal{D}$ generate adversarial perturbations $\delta_i^{(t)}$ given the current model parameters $\theta^{(t)}$ and perturbation budget ϵ

2. Augument training set (max. problem)

Define $\mathcal{D}_{adv} = \{(x_i + \delta_i^{(t)}, y_i) | x_i \in \mathcal{D}'\}$ and construct a new training set $\mathcal{D}^{(t)} = \mathcal{D}_{adv} \cup \mathcal{D}$

3. Adversarial training (min. problem)

Train on augumented $\mathcal{D}^{(t)}$ for $N \geq 1$ epochs, then repeat from Step 1.

Problems of conventional AT (1)

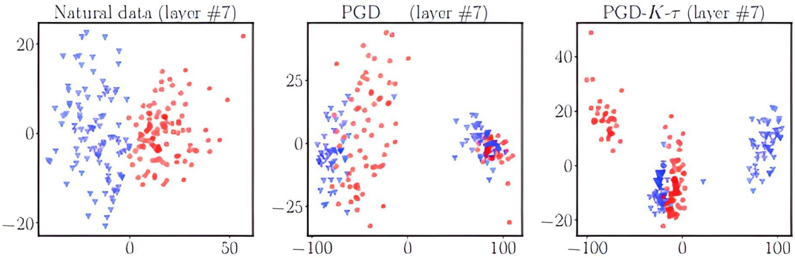
Problem: Adversarial perturbations generated by strong attacks significantly cross over the decision boundary and are close to natural data, which hurts the generalization.

Paper: Attacks Which Do Not Kill Training Make Adversarial Learning Stronger

Solution: *Friendly Adversarial Training* (FAT) – instead of finding the worst-case attack, search for the weakest one to construct $\mathcal{D}_{adv}$

Problems of conventional AT (1)

An *Early-stopped PGD* algorithm ($PGD - K - \tau$) is used to find weak adversarial samples without affecting generalization.



Feature projections for two classes (▽ and ▽). While standard PGD leads to a significant mixing of the two classes, $PGD - K - \tau$ maintains separability, i.e. generalization. [Source]

Another idea: just use an attack which searches for the minimal perturbation (e.g. DDN, FMN)

Problems of conventional AT (2)

Problem: Individual data points have different intrinsic robustness – different distances to the decision boundary. However, AT uses the same fixed ϵ to generate adversarial data from all of them.

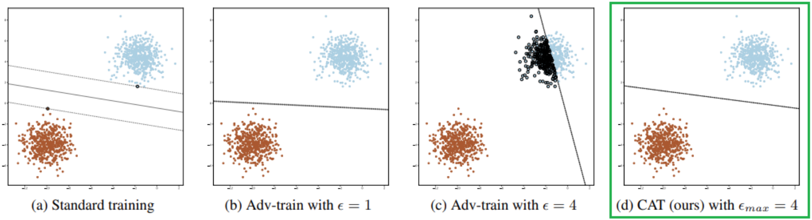
Paper: CAT: *Customized Adversarial Training for Improved Robustness*

Solution: Adaptively modify the ϵ_i for the i^{th} training sample:

$$\epsilon_i = \underset{\epsilon}{\operatorname{argmin}} \left\{ \max_{\boldsymbol{x}_i' \in \mathcal{B}(\boldsymbol{x}_i, \epsilon)} f_\theta\left(\boldsymbol{x}_i'\right) \neq y_i \right\}$$

The inner maximization uses PGD to find $\boldsymbol{x}_i'$, while the outer minimization modifies ϵ_i at each epoch depending on the attack's success.

Problems of conventional AT (2)



Decision boundaries for a linearly separable binary classification problem. [Source]
(b) AT with low fixed ϵ leads to a decision boundary close to some examples.
(c) AT with high fixed ϵ leads to loss of generalization, adversarial samples becoming mixed with the other class.
(d) CAT with bounded flexible ϵ leads to a good trade-off between generalization and robustness

AT + Fake Data

Problem: AT suffers from *Robust Overfitting* – **robust test accuracy** starts decreasing during training, while **robust train accuracy** continues to rise

Paper: *Overfitting in adversarially robust deep learning*

Why does it happen?

- Network gets used to the *adversarial noise* generated for training data
- But loses generality when it comes to *adversarial noise* generated on test data (i.e., it learns the noise rather than the pattern)

AT + Fake Data

Problem: AT suffers from *Robust Overfitting* – **robust test accuracy** starts decreasing during training, while **robust train accuracy** continues to rise

Paper: *Overfitting in adversarially robust deep learning*

Why does it happen?

- Network gets used to the *adversarial noise* generated for training data
- But loses generality when it comes to *adversarial noise* generated on test data (i.e., it learns the noise rather than the pattern)

AT + Fake Data

Problem: AT suffers from *Robust Overfitting* – **robust test accuracy** starts decreasing during training, while **robust train accuracy** continues to rise

Paper: *Overfitting in adversarially robust deep learning*

Why does it happen?

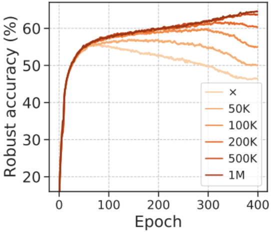
- Network gets used to the *adversarial noise* generated for training data
- But loses generality when it comes to *adversarial noise* generated on test data (i.e., it learns the noise rather than the pattern)

AT + Fake Data

Solution: Replace adversarial training samples with synthetically generated data (e.g. DDPM, FM)

Papers:

- *Fixing Data Augmentation to Improve Adversarial Robustness*
- *Better Diffusion Models Further Improve Adversarial Training*



Robust accuracy evolution when using different amounts of generated data during AT. 'x' corresponds to no additional data. [Source]

AT + Fake Data

Augument training set with generated data

Using a set of generators $\mathcal{G} = \{g_1, \ldots, g_m\}$, define the augumented training set $\mathcal{D}_{aug} = \mathcal{D}_{fake} \cup \mathcal{D}$ as follows:

- *unconditional* generation:

$$\mathcal{D}_{fake} = \left\{ \left(g_i(z), \mathcal{T}\left(g_i(z)\right) \right) \middle| g_i \in \mathcal{G}, z \sim \mathcal{N}(0,1) \right\}$$

  where $\mathcal{T}(\cdot)$ is a pre-trained classifier on $\mathcal{D}$

- *conditional* generation:

$$\mathcal{D}_{fake} = \left\{ \left(g_i(z|\, y_j), y_j \right) \middle| g_i \in \mathcal{G}, z \sim \mathcal{N}(0,1), y_j \in \mathcal{Y} \right\}$$

  where $\mathcal{Y}$ is the set of all labels in $\mathcal{D}$

Regularization Techniques

Regularization Training

Idea : Use an objective for minimization that leads to robust structures, while maintaining clean accuracy

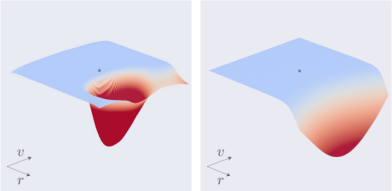
Including additional loss terms

$$\mathcal{L}(x_i, y_i, \theta) \leftarrow \mathcal{L}_{cls}(x_i, y_i, \theta) + \underbrace{\alpha \mathcal{L}_{reg}(x_i, y_i, \theta, \texttt{*args})}_{\text{pushes towards a desired direction}}$$

- θ – network parameters
- $\mathcal{L}_{cls}$ – standard loss
- α – regularization factor
- $\mathcal{L}_{reg}$ – regularization loss
- `*args` – method-specific hyperparameters

Regularization Training (1)

Motivation: AT leads to a significant decrease in the curvature of the loss surface with respect to inputs, leading to a more "linear" behavior



(a) Original (CIFAR-10) (b) Fine-tuned (CIFAR-10)

Loss landscape before AT (a) and after AT (b). Blue color corresponds to low loss values, while red corresponds to high loss $\approx$ adversarial regions

Paper : Robustness via Curvature Regularization, and Vice Versa

Idea : Penalize large eigenvalues in the Hessian of loss w.r.t. input points.

Regularization Training (1)

Curvature Regularization

$$\mathcal{L}_{reg}(x, y, \theta) = \|\nabla \mathcal{L}_{cls}(x + hz, y, \theta) - \nabla \mathcal{L}_{cls}(x, y, \theta)\|_2^2$$

where h is a small constant, and:

$$z = \frac{\nabla_x \mathcal{L}_{cls}(x, y, \theta)}{\|\mathcal{L}_{cls}(x, y, \theta)\|_2}, \quad \text{OR} \quad z \sim \mathcal{N}(0, \mathbf{I})$$

In the second case, an expectation of $\mathcal{L}_{reg}$ over z is used.

- $\mathcal{L}_{cls}$ is a standard classification loss
- Different from AT which flattens the whole loss landscape around clean samples, this reduces curvature only in some significant directions, where adversarial attacks are more likely to be found

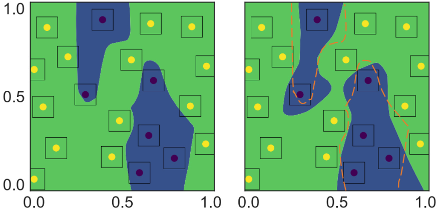
Regularization Training (2)

Paper : TRADES: TRadeoff-inspired Adversarial DEfense via Surrogate-loss minimization

Idea: Manipulate the trade-off between natural error $\mathcal{R}_{nat}$ and robust error $\mathcal{R}_{rob}$, where:

$$\mathcal{R}_{rob} = \mathcal{R}_{nat} + \mathcal{R}_{bdy}$$

where $\mathcal{R}_{bdy}$ is the *boundary error*, measuring the n.o. correctly classified points laying near the decision boundary



Standard training (**left**) and TRADES (**right**) decision boundaries.

Regularization Training (2)

TRADES

$$\mathcal{L}_{reg}(x, y, \theta) = \max_{x',\, |x'-x| \leq \epsilon} \mathcal{L}_{CE}\big(f_\theta(x'), f_\theta(x)\big)$$

where $\mathcal{L}_{CE}$ is the standard cross-entropy loss

- Encourages the output to be smooth in the vicinity of x, pushing the decision boundary away
- Similarity with AT: forces adversarial samples x' to be *seen* as clean samples x by the network f_θ
- Difference from AT: forces the entire output distribution to be the same

Regularization Training (3)

Paper : Metric Learning for Adversarial Robustness

Idea : Introduces *Triplet Loss Adversarial* (TLA) learning, pushing towards:

1. Similarity between adversarial examples x_a and samples they've been generated from x_p (*positive*)
2. Dissimilarity between adversarial examples x_a and clean samples from a different class x_n (*negative*)

Regularization Training (3)

TLA – Triplet loss regularizer

$$\mathcal{L}_{reg}(x_p, x_a, x_n, \theta) = \max\left(D\big(f_\theta^{(k)}(x_p), f_\theta^{(k)}(x_a)\big) - D\big(f_\theta^{(k)}(x_n), f_\theta^{(k)}(x_a)\big) + \alpha, 0\right)$$

where $x_a = x_p + \delta$, $f_\theta^{(k)}$ outputs the feature vector from k^{th} layer, and:

$$D(x, y) = 1 - \frac{|\langle x, y\rangle|}{\|x\|_2 \|y\|_2}$$

- Hyperparameter α is called *margin*, and it forces the two distances to be at least α-distanced
- x_a is called *anchor* since all distances are related to it
- Negative samples x_n for computing $\mathcal{L}_{reg}$ are selected based on the closest negatives to x_a, measured in the embedding space

Regularization Training (3)

Problem : Features $f_\theta^{(k)}$ might have very different magnitude ranges between x_a, x_p, x_n

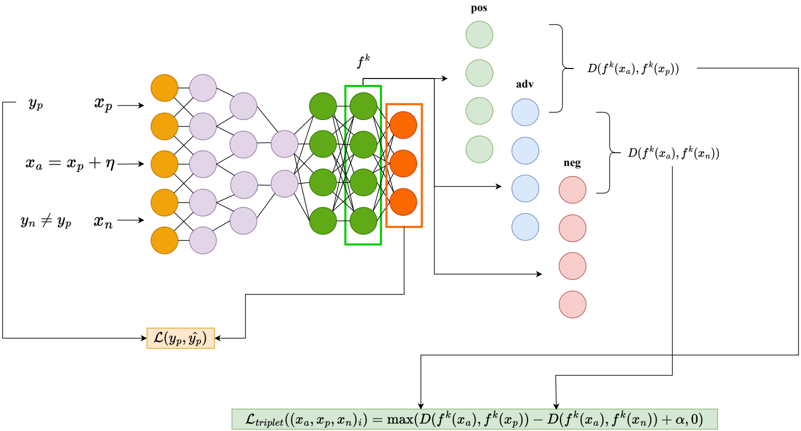
Solution :

TLA with Norm Regularization

$$\mathcal{L}_{reg}(x_p, x_a, x_n, \theta) + \lambda\Big(\|f_\theta^{(k)}(x_a)\|_2 + \|f_\theta^{(k)}(x_p)\|_2 + \|f_\theta^{(k)}(x_n)\|_2\Big)$$

- Further reduces the influence of very high / low feature norms over the regularization term
- Forces similarity / dissimilarity in the *angular space*, rather than ℓ_2 radius

Regularization Training (3)

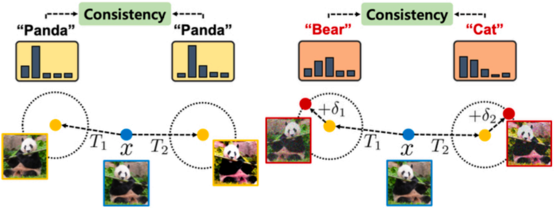


Tiplet Loss Adversarial training workflow

Regularization Training (4)

Paper : Consistency Regularization for Adversarial Robustness

Idea : Enforce consistency between adversarial examples generated from different transformations of the same image



(a) Conventional CR (b) Proposed CR

Usual consistency (a) and *attacked augumentation* consistency (b). T_1 and T_2 are two different image transformations (e.g. rotation, random crop, flip).

Regularization Training (4)

Consistency Regularization

$$\mathcal{L}_{reg} = \texttt{JS}\Big(\hat{f}_\theta\big(T_1(x) + \delta_1, \tau\big) \,\|\, \hat{f}_\theta\big(T_1(x) + \delta_1, \tau\big)\Big)$$

where JS is the Jensen-Shannon divergence (i.e. symmetric variant of KL) and

$$\hat{f}_\theta\big(T(x) + \delta, \tau\big) = \texttt{Softmax}\Big(\frac{f_\theta\big(T(x) + \delta\big)}{\tau}\Big)$$

is a temperature-scaled (τ) variant of classifier f_θ

- Augumentations are randomly sampled in every training step $\rightarrow$ prediction on adversarial data becomes consistent
- Temperature-scaling enforces a sharper distribution ($\tau \in (0, 1)$), since confidence (i.e. max probability) for AT remains low

Certified Robust Methods

Why "certified"?

Problems :

- AT and Regularization training are empirical defenses, which aim to improve robustness against empirical attacks
- We need mechanisms for providing / imposing a lower bound on robust accuracy, against **any attack** with maximum budget ϵ

Paper : Certified Defenses Against Adversarial Eamples

Solution : **Certified Defenses:**

1. Probabilistic
2. Deterministic

Why "certified"?

Certified Defense formulation

For a given input x and it's associated class y, a certification assessment over a robust classifier f_θ w.r.t. an attack budget ϵ can be stated as follows:

$$\arg\max_j f_\theta(x+\delta)_j = y, \quad \forall\delta \text{ with } \|\delta\|_p \leq \epsilon$$

Note that the above statement doesn't refer to any particular attack mechanism. More concretely, the following statement between logits holds:

$$f_\theta(x+\delta)_y - f_\theta(x+\delta)_j \geq m(\epsilon,\theta), \quad \forall j \neq y$$

where $m(\epsilon,\theta)$ is called the *certified margin*. If $m(\epsilon,\theta) > 0$, classifier f_θ is certified robust within $\mathbb{B}_p(x,\epsilon) = \{x' \mid \|x-x'\|_p \leq \epsilon\}$.

Core tasks :

- Construct effective and accurate margins $m(\epsilon)$
- Design algorithms for controlling these margins

Certified Probabilistic Defenses (1)

Papers :

- Certified Adversarial Robustness via Randomized Smoothing **(RS)**
- Randomized Smoothing of All Shapes and Sizes

Idea : Construct a *smoothed* classifier g_θ from a base classifier f_θ, s.t.:

$$g_\theta(x, \sigma) = \arg\max_c \mathbb{P}\big(\arg\max_j f_\theta(x + \epsilon)_j = c\big),\ \epsilon \sim \mathcal{N}(0, \sigma^2 I)$$

which corresponds to the most likely class for a Gaussian perturbed x, and σ is a hyperparameter controlling the robustness / accuracy trade-off.

How do we assess the ℓ_2 robustness radius around x?

Certified Probabilistic Defenses (1)

Robustness Guarantee of RS

Suppose that for input data sampled from $\mathcal{N}(x, \sigma^2 I)$, the most probable class predicted by f_θ is c_A with probability p_A, and the second most likely class is c_B with probability p_B. Then, the smoothed classifier g_θ is robust around x with ℓ_2 radius:

$$R(x, g_\theta) = \frac{\sigma}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right)$$

which corresponds to:

$$\arg\max f_\theta(x+z) = \arg\max f_\theta(x), \; \forall \|z\|_2 \leq R$$

where

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{0} e^{-t^2/2} dt$$

is the CDF of a standard Gaussian.

- This provides a link between input Gaussian noise power and certified ℓ_2 distance

Certified Probabilistic Defenses (1)

Prediction & Certification

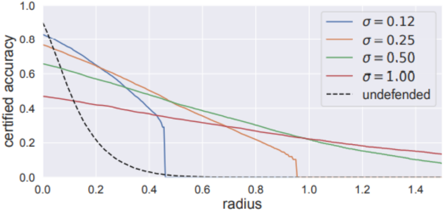
1. Sample N points from $\mathcal{N}(x, \sigma^2 I)$ and compute predictions of f_θ
2. Compute the two most frequent classes c_A and c_B, and their corresponding counts n_A and n_B
3. Compute the p-value of binomial test (for $p = 0.5$ and significance level α):

$$\texttt{p-val} = \sum_{k=n_A}^{n_A+n_B} \binom{n_A + n_B}{k} p^k (1-p)^{n_A+n_B-k}$$

   1. if $\texttt{p-val} > \alpha$: **ABSTAIN** (accept null hypothesis $p = 0.5$ and deem the result statistically insignificant)
   2. else:
      - **RETURN** c_A as the robust prediction
      - Estimate a lower bound $\underline{p_A}$ and upper bound $\overline{p_B}$ and **RETURN** R

Certified Probabilistic Defenses (1)

- **Training**: Gaussian data augumentation at variance σ^2



Certified accuracy of RS on CIFAR-10 udner different train settings. [Source]

- **Drawbacks :**
  - Requires many samples to estimate certified radius (e.g. $N = 10^5$)
  - Can't impose a specific certified radius, since everything is randomly constructed

Certified Probabilistic Defenses (2)

Problem : How to maximize the ℓ_2 robustness without explicitly adding ℓ_2 adversarial perturbations to data (i.e. AT) ?

Paper : MACER: Attack-free and scalable robust training via maximizing certified radius

Solution :

- Attack-free training
- Maximizes ℓ_2 certified radius during training, using Gaussian data augumentation and an additional loss term (regularization)

Certified Probabilistic Defenses (2)

MACER – MAximizing the CErtified Radius

$$\mathcal{L}(x, y, \theta) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{cls}(x + \eta_i, y, \theta) + \underbrace{\frac{\lambda\sigma}{2} \max\left(\gamma - \hat{R}(x, g_\theta), 0\right)\Big|_{g_\theta(x)=y}}_{\ell_2 \text{ radius maximization}}$$

where $\hat{R}$ is a **soft randomized smoothing radius** version of R, which can be differentiated w.r.t. θ

- The first term corresponds to RS training
- The second term pushes certified radix to be close to γ, for points correctly classified by smoothed classifier g_θ

Certified Deterministic Defenses (1)

Problem : How to directly control the output perturbation given the input one?

Papers :

- Achieving robustness in classification using optimal transport with hinge regularization
- Pay attention to your loss: understanding misconceptions about 1-Lipschitz neural networks

Solution : Structurally constrain neural networks to impose a desired Lipschitz bound $L_{p,q}$ over f_θ:

$$L_{p,q} = \max_{x \neq y} \frac{\|f_\theta(x) - f_\theta(y)\|_p}{\|x - y\|_q}, \quad p, q \in \mathbb{N}$$

This ensures that:

$$\|f_\theta(x + \delta) - f_\theta(x)\|_p \leq L_{p,q} \|\delta\|_q$$

Certified Deterministic Defenses (1)

Lipschitz bound of feed-froward networks

Consider f_θ to be a feed-forward neural network with no skip connections, 1-Lipschitz activation functions, and m layers defined by their weight tensors $\{W_i\}_{i=1}^m$. Then, for the usual case $p = q = 2$ the following holds:

$$L_{2,2} \leq \prod_{i=1}^{m} \|W_i\|_{2,2}$$

which is an upper bound of the Lipschitz constant of f_θ, and

$$\|W_i\|_{p,q} = \max_{x \neq 0} \frac{\|W_i x\|_p}{\|x\|_q}$$

- For 2D convolutional layers $W_i \in \mathbb{R}^{C_{in} \times C_{out} \times h \times w}$ one has the following options:
  1. Reshape to $\widetilde{W}_i \in \mathbb{R}^{hwC_{in} \times C_{out}}$ and compute spectral norm $\|\widetilde{W}_i\|_2$
  2. Compute 2D FFT $\mathbf{W_i} \in \mathbb{R}^{C_{in} \times C_{out} \times n_{fft} \times n_{fft}}$ and take the maximum magnitude over frequency dimensions (roughly)
  3. Other techniques ...

Certified Deterministic Defenses (1)

Imposing Lipschitz bounds – Spectral Normalization

For a network f_θ as previously defined, one can impose a Lipschitz bound $L_{2,2} = \hat{L}$ using the following normalization scheme:

$$W_i' \leftarrow \frac{W_i}{\| W_i \|_2} (\hat{L})^{\frac{1}{m}}$$

which holds $\prod_{i=1}^{m} \| W_i' \|_2 = \hat{L}$

- Spectral Normalization can be applied as an additional step after the usual gradient update
- Faster techniques such as *Power Iteration* and Gram Iteration are employed to efficiently estimate $\| W_i \|_2$ during training

Certified Deterministic Defenses (1)

Problem : In a classification setting, we're interested in the ℓ_2 norm of each output neuron, not the entire logit vector

Lipschitz bounds of individual logits

For a network f_θ as previously defined, the Lipschitz bound of logit f_θ^j is defined as follows:

$$L_{2,2}^j \leq \left(\prod_{i=1}^{m-1} \|W_i\|_2\right) \|W_m[:,j]\|_2$$

where $W_m[:,j]$ corresponds to the j^{th} column of W_m, connected to the j^{th} logit

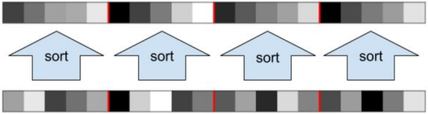
- $\|W_m[:,j]\|_2$ can be imposed similarly to the other layers
- We can quantify how much the correct logit y will change when input is perturbed:

$$\|f_\theta^y(x+\delta) - f_\theta^y(x)\|_2 \leq L_{2,2}^y \|\delta\|_2$$

Certified Deterministic Defenses (1)

Requirements for training 1-Lipschitz networks (under the previous settings)

- **Norm-Preserving activation functions**, e.g. `GroupSort` / `FullSort`:



Source: Sorting Out Lipschitz Function Approximation

- **Gradient Norm Preservation** during backpropagation:
  Bjorck orthonormalization over constrained W_i's $\rightarrow$
  iteratively finds the closest orthonormal matrix $\rightarrow$
  imposes all singular values to be close to 1, to help gradient flow

Certified Deterministic Defenses (2)

Papers :

- Globally-Robust Neural Networks (GloRoNets)
- Relaxing Local Robustness

Idea : Introduce an additional class (denoted $\perp$) to signal that a point cannot be certified as ϵ-**globally-robust**

Local vs Global ϵ-robustness

Local (at x): $\|x - x'\| \leq \epsilon \implies \text{arg}\max_j f_\theta(x)_j = \text{arg}\max_j f_\theta(x')_j, \forall\, x'$

Global: $\|x_1 - x_2\| \leq \epsilon \implies \text{arg}\max_j f_\theta(x_1)_j \overset{\perp}{=} \text{arg}\max_j f_\theta(x_2)_j, \forall\, x_1, x_2$

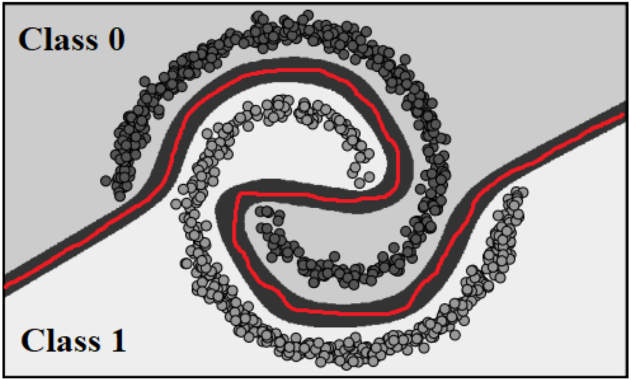
,where $\overset{\perp}{=}$ holds `True` if:

$$\text{arg}\max_j f_\theta(x_1)_j = \perp \quad \texttt{OR}$$

$$\text{arg}\max_j f_\theta(x_2)_j = \perp \quad \texttt{OR}$$

$$\text{arg}\max_j f_\theta(x_1)_j = \text{arg}\max_j f_\theta(x_2)_j$$

Certified Deterministic Defenses (2)



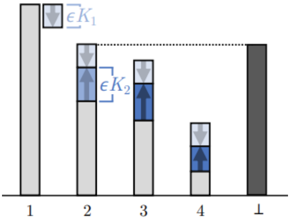
Global robustness between two classes. Red line corresponds to the decision boundary, with surrounding dark gray area corresponding to points labeled as $\perp$, due to closeness to the boundary. [Source]

Certified Deterministic Defenses (2)

Constructing logit $\perp$

Denote $K_i = L^i_{2,2}$ the Lipschitz constant of the i^{th} logit. Then, for an example x correctly classified as $y = \arg\max_j f_\theta(x)_j$, the logit $y_\perp$ is constructed as follows:

$$y_\perp = \max_{i \neq y}\{f_\theta(x)_i + (K_i + K_y)\epsilon\}$$



$\perp$ logit accounts for two worst-case scenarios:

1. Predicted class decreases by maximum amount ϵK_1
2. Second-most likely class increases by maximum amount ϵK_2

The model is pushed to predict logits for the correct class that surpass logit $\perp \implies$ global $\epsilon-$robustness

Certified Deterministic Defenses (2)

Training GlRoNets

1. For each element x_i in batch B construct logit value $y_\perp$, with an initially set ϵ
2. Concatenate $y_\perp$ with $f_\theta(x_i)$ and optimize network over this augumented logit vector

- For a GloRoNet trained with a specific ϵ, it's final accuracy corresponds to the fraction of points which are globally $\epsilon-$robust:

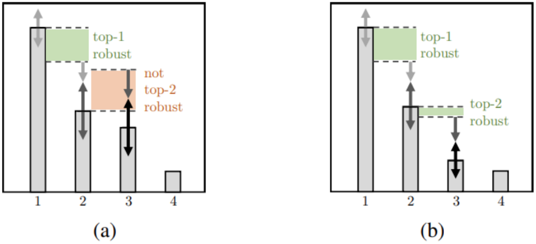
$$\forall \delta \text{ with } \|\delta\| \leq \epsilon \quad \text{and} \quad \forall x \text{ correctly classified,} \quad x+\delta \quad \text{is correctly classified}$$

- Alternatively, for any new point x we can say what's the highest ϵ for which it's certified that any attack with $\|\delta\| \leq \epsilon$ won't work

Observation: GloRoNets are shown to impose strong implicit regularization on global Lipschitz.
Note: Lipschitz bound needs to be computed each time $y_\perp$ is computed. Can it be made more efficient?

Certified Deterministic Defenses (2)

Generalisation : Certified top-k robustness



Top-2 robust (b) vs. non-top-2 robust (a) predictions. In (a), the third logit is sufficiently large to overcome the second, if sufficiently perturbed. [Source]

- We're interested in keeping the top-k predictions unchanged
- Logit $\perp$ is constructed s.t. the minimum logit change that can change the top-k predictions is pushed away

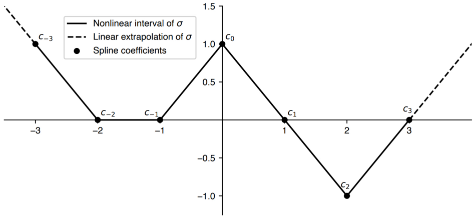
Certified Deterministic Defenses (3)

Problem : Common activation functions limit the expressivity of Lipschitz-constrained networks (e.g. `ReLU`, `LeakyReLU`)

Paper : Improving Lipschitz-Constrained Neural Networks by Learning Activation Functions

Solution : Define a controllable structure for activation functions, ensure universal approximation theory, and make them learnable:

$$\sigma_{\ell,n}(x) = b_{1,\ell,n} + b_{2,\ell,n}x + \sum_{k=1}^{K_{\ell,n}} a_{k,\ell,n}\,\mathrm{ReLU}\left(x - \tau_{k,\ell,n}\right)$$



Example of Learnable Linear Spline (LLS) activation. [Source]

Certified Defenses – Extending to Multi-modal inputs

Problem : Hard to adapt previous methods for complex networks / tasks, in a certified manner

Paper : MMCert: Provable Defense against Adversarial Attacks to Multi-modal Models

- Input of T modalities: $M = (m_1, m_2, \dots, m_T)$, $|m_i| = n_i$
- Attack types: modification, addition, deletion
- N.o. maximum elements attacked in each modality: $R = (r_1, r_2, \dots, r_T)$
- Sub-sampled input: $Z = (z_1, z_2, \dots, z_T)$, $|z_i| = k_i$

Idea : Use Monte Carlo to estimate a lower and an upper bound for the probability of correct class over multiple sub-sampled inputs:

$$\underline{p_y} = \text{LB}(f_\theta(Z)); \quad \overline{p_y} = \text{UB}(f_\theta(Z))$$

Certified Defenses – Extending to Multi-modal inputs

Certified Multi-Modal Classification

Consider N sub-samplings $Z_1, \ldots, Z_N$ and amulti-modal classifier f_θ.

1. Compute predictions over all Z_i
2. Compute top-2 most frequent classes A and B and their lower and upper bounds $\underline{p_A}$ and $\overline{p_B}$, using RS techniques
3. Compute: $\delta_l = \underline{p_A} - \frac{\left\lfloor \underline{p_A} \prod_{i=1}^{T} \binom{n_i}{k_i} \right\rfloor}{\prod_{i=1}^{T} \binom{n_i}{k_i}}$ and $\delta_u = \frac{\left\lceil \overline{p_B} \prod_{i=1}^{T} \binom{n_i}{k_i} \right\rceil}{\prod_{i=1}^{T} \binom{k_i^i}{k_i}} - \overline{p_B}$
4. Then, f_θ is robust w.r.t. $R = (r_1, r_2, \ldots, r_T)$ if:

$$\frac{\prod_{i=1}^{T} \binom{n_i}{k_i}}{\prod_{i=1}^{T} \binom{n'_i}{k_i}} \left(\underline{p_A} - \delta_l - 1 + \frac{\prod_{i=1}^{T} \binom{e_i}{k_i}}{\prod_{i=1}^{T} \binom{n_i}{k_i}} \right) \geq \frac{\prod_{i=1}^{T} \binom{n_i}{k_i}}{\prod_{i=1}^{T} \binom{n'_i}{k_i}} (\overline{p_B} + \delta_u) + 1 - \frac{\prod_{i=1}^{T} \binom{e_i}{k_i}}{\prod_{i=1}^{T} \binom{n'_i}{k_i}}$$

where $e_i = n_i - r_i$

Note: For segmentation (i.e. multi-classification) things become a lot more complicated.

Certified Deterministic Defenses – Limitations

- Reduction in clean accuracy:

Imposing hard constraints affects prediction on points close to the decision boundary.

- Bounds for complex structures:

Developing tight Lipschitz bounds for complex networks (e.g. Transformers) is not straightforward.

- Computational overhead

Certified methods require additional computational steps compared to to standard training.