ADVERSARIAL ROBUSTNESS THROUGH RANDOMIZATION

Lucas GNECCO HEREDIA

CNRS, LAMSADE - Université Paris Dauphine - PSL

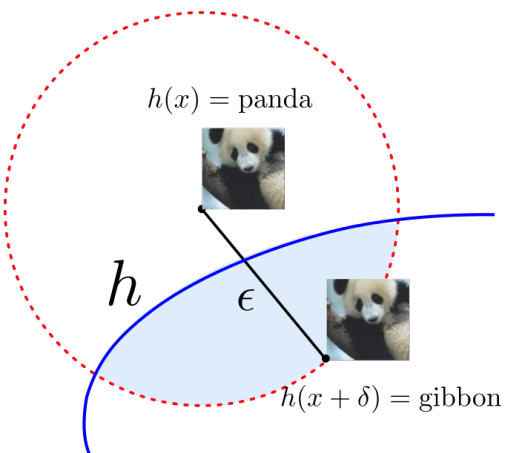
November 22, 2023

Table of Contents

Table of Contents

Problem setting: Adversarial classification

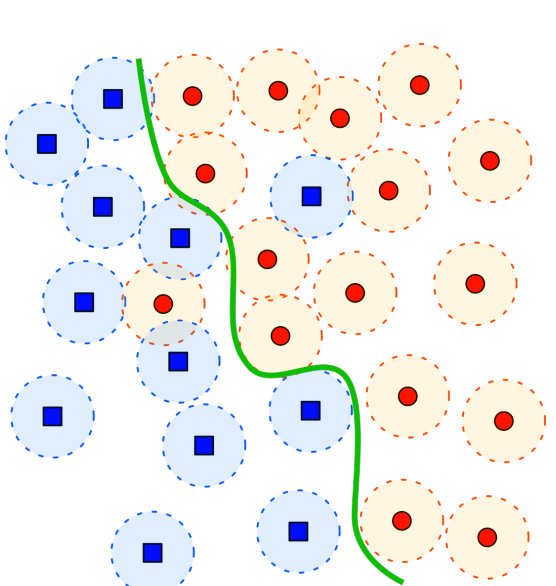
- Data perturbing adversary with budget ϵ can transport any x to $x' \in B_\epsilon(x) = \{x' \in \mathcal{X} \mid d(x, x') \leq \epsilon\}$ to induce an error.

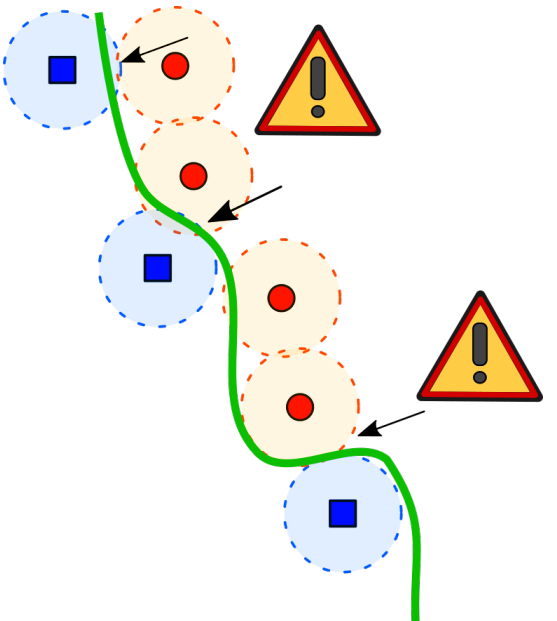


Goal Find $h : \mathcal{X} \to \mathcal{Y}$ within some family $\mathcal{H}$ with the lowest **adversarial** risk (highest **robust** accuracy)

$$\mathcal{R}_\epsilon(h) = \mathbb{E}_{(x,y)\sim\rho}\left[\sup_{x' \in B_\epsilon(x)} \ell^{0\text{-}1}((x', y), h)\right]$$

Visualization (adversarial classification)





RANDOMIZED CLASSIFIERS IN THE LITERATURE

Many previous works have proposed *stochastic* or *randomized* models as a way to improve robustness to adversarial attacks.

Randomized classifiers in theory

Intuitively, the output of a **randomized classifier** is not a label, but a *probability distribution* over labels.

Randomized classifiers in theory

Intuitively, the output of a **randomized classifier** is not a label, but a *probability distribution* over labels.

- Randomized: $\mathbf{h} : \mathcal{X} \to \Delta^K$.
- Deterministic: $h : \mathcal{X} \to \{1, \dots, K\} \cong \{e_1, \dots, e_K\} \subset \Delta^K$.

Randomized classifiers in practice

In practice, randomized classifiers involve randomized transformations of the **input** or **model**.

- Input noise injection [HRF19; Pin+19; Yu+21]

$$x \to \boxed{\text{sample noise } \eta \sim \mu} \to h(x + \eta)$$

- Weight noise injection or model sampling [HRF19; Pin+20; DS22; Wic+21; Dhi+18]

$$x \to \boxed{\text{sample model } h \sim \mu} \to h(x)$$

Randomized classifiers in practice

In practice, randomized classifiers involve randomized transformations of the **input** or **model**.

- Input noise injection [HRF19; Pin+19; Yu+21]

$$x \to \boxed{\text{sample noise } \eta \sim \mu} \to h(x + \eta)$$

- Weight noise injection or model sampling [HRF19; Pin+20; DS22; Wic+21; Dhi+18]

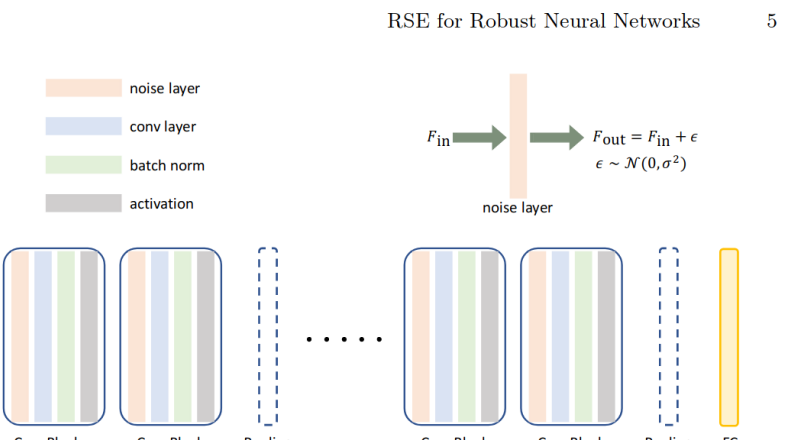
$$x \to \boxed{\text{sample model } h \sim \mu} \to h(x)$$

Most methods can be though as a distribution over some family models...

Table of Contents

RANDOM SELF ENSEMBLE **[ECCV 2018]** [LIU+18]

Basically Noise layers + Avg prediction



RANDOM SELF ENSEMBLE **[ECCV 2018]** [LIU+18]

Algorithm 1 Training and Testing of Random Self-Ensemble (RSE)

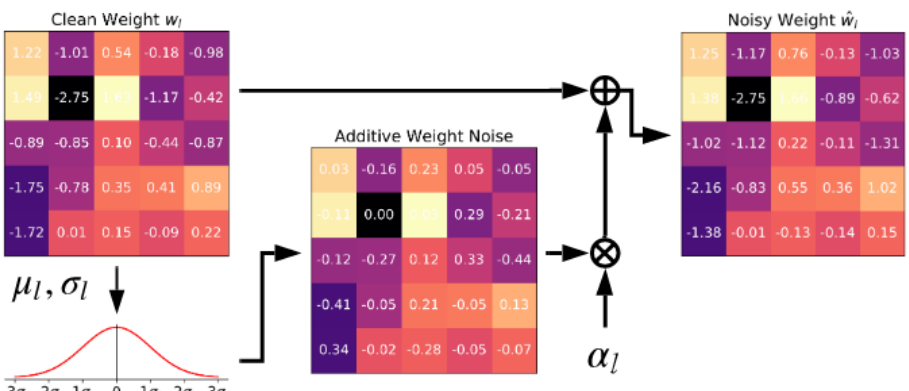
Training phase:
for iter $= 1, 2, \ldots$ **do**
  Randomly sample (x_i, y_i) in dataset
  Randomly generate $\epsilon \sim \mathcal{N}(0, \sigma^2)$ for each noise layer.
  Compute $\Delta w = \nabla_w \ell(f_\epsilon(w, x_i), y_i)$ (Noisy gradient)
  Update weights: $w \leftarrow w - \Delta w$.
end for
Testing phase:
Given testing image x, initialize $p = (0, 0, \ldots, 0)$
for $j = 1, 2, \ldots, \#\text{Ensemble}$ **do**
  Randomly generate $\epsilon \sim \mathcal{N}(0, \sigma^2)$ for each noise layer.
  Forward propagation to calculate probability output

$$p^j = f_\epsilon(w, x)$$

  Update p: $p \leftarrow p + p^j$.
end for
Predict the class with maximum score $\hat{y} = \arg\max_k p_k$

Parametric Noise Injection [CVPR 2019] [HRF19]

Weight or input noise injection + Adv training.



Parametric Noise Injection [CVPR 2019] [HRF19]

Noise intensity is learnable. For layer l and weight i, the noised version is

$$\tilde{v}_{l,i} = f_{PNI}(v_{l,i}) = v_{l,i} + \alpha_l \cdot \eta_{l,i},\ \eta_{l,i} \sim \mathcal{N}(0, \sigma_l^2).$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_l} = \sum_i \frac{\partial \mathcal{L}}{\partial f_{PNI}(v_{l,i})} \underbrace{\frac{\partial f_{PNI}(v_{l,i})}{\partial \alpha_l}}_{\eta_{l,i}}$$

Activation Pruning [ICLR 2018] [Dhi+18]

Algorithm 1 Stochastic Activation Pruning (SAP)

Input: input datum x, neural network with n layers, with i^{th} layer having weight matrix W^i, non-linearity ϕ^i and number of samples to be drawn r^i.
$h^0 \leftarrow x$
for each layer i **do**
  $h^i \leftarrow \phi^i(W^i h^{i-1})$ ▷ activation vector for layer i with dimension a^i
  $p^i_j \leftarrow \frac{|(h^i)_j|}{\sum_{k=1}^{a^i} |(h^i)_k|}, \forall j \in \{1, \ldots, a^i\}$ ▷ activations normalized on to the simplex
  $S \leftarrow \{\}$ ▷ set of indices not to be pruned
  repeat r^i **times** ▷ the activations have r^i chances of being kept
    Draw $s \sim \text{categorical}(p^i)$ ▷ draw an index to be kept
    $S \leftarrow S \cup \{s\}$ ▷ add index s to the keep set
  for each $j \notin S$ **do**
    $(h^i)_j \leftarrow 0$ ▷ prune the activations not in S
  for each $j \in S$ **do**
    $(h^i)_j \leftarrow \frac{(h^i)_j}{1-(1-p^i_j)^{r^i}}$ ▷ scale up the activations in S
return h^n

Other approaches

- Random resize and padding [Xie+17]
- Stochastic Local-Winner-Takes-All [PCT21; Pan+21]
- Simple and Effective Stochastic Neural Networks [Yu+21]

Obfuscated gradients

Many (if not all) of the methods do not provide *real* robustness. They just make it harder to find an attack with the usual gradient methods [ACW18].

Defense	Dataset	Distance	Accuracy
Buckman et al. (2018)	CIFAR	0.031 (ℓ_∞)	0%*
Ma et al. (2018)	CIFAR	0.031 (ℓ_∞)	5%
Guo et al. (2018)	ImageNet	0.005 (ℓ_2)	0%*
Dhillon et al. (2018)	CIFAR	0.031 (ℓ_∞)	0%
Xie et al. (2018)	ImageNet	0.031 (ℓ_∞)	0%*
Song et al. (2018)	CIFAR	0.031 (ℓ_∞)	9%*
Samangouei et al. (2018)	MNIST	0.005 (ℓ_2)	55%**
Madry et al. (2018)	CIFAR	0.031 (ℓ_∞)	47%
Na et al. (2018)	CIFAR	0.015 (ℓ_∞)	15%

On Adaptive Attacks to Adversarial Example Defenses [Tra+20]

On Adaptive Attacks to Adversarial Example Defenses

Florian Tramèr*
Stanford University
tramer@cs.stanford.edu

Nicholas Carlini*
Google
nicholas@carlini.com

Wieland Brendel*
University of Tübingen
wieland.brendel@uni-tuebingen.de

Aleksander Mądry
MIT
madry@mit.edu

Table of Contents

EXPECTED RISK

Suppose that the randomness of the model can be described by some distribution μ over a family of classifiers $\mathcal{H}$.

$$x \to \boxed{\text{sample model } h \sim \mu} \to h(x)$$

EXPECTED RISK

Deterministic	Randomized
$\mathcal{R}(h) = \mathbb{E}_{x,y}[\ell(h, x, y)]$	$\mathcal{R}(\mathbf{h}_\mu) = \mathbb{E}_{x,y}\mathbb{E}_{h\sim\mu}[\ell(h, x, y)]$
$\mathcal{R}_\epsilon(h) = \mathbb{E}_{x,y}[\ \sup_{x' \in B_\epsilon(x)}\ \ell(h, x', y)]$	$\mathcal{R}_\epsilon(\mathbf{h}_\mu) = \mathbb{E}_{x,y}\left[\ \sup_{x' \in B_\epsilon(x)}\ \mathbb{E}_{h\sim\mu}[\ell(h, x', y)]\right]$

EXPECTED RISK

Deterministic	Randomized
$\mathcal{R}(h) = \mathbb{E}_{x,y}[\ell(h, x, y)]$	$\mathcal{R}(\mathbf{h}_\mu) = \mathbb{E}_{x,y}\mathbb{E}_{h\sim\mu}[\ell(h, x, y)]$
$\mathcal{R}_\epsilon(h) = \mathbb{E}_{x,y}[\ \sup_{x' \in B_\epsilon(x)}\ \ell(h, x', y)]$	$\mathcal{R}_\epsilon(\mathbf{h}_\mu) = \mathbb{E}_{x,y}\left[\ \sup_{x' \in B_\epsilon(x)}\ \mathbb{E}_{h\sim\mu}[\ell(h, x', y)]\right]$

We will focus on the case of the 0-1 loss

$$\ell^{0\text{-}1}(h, x, y) = \mathbb{1}\left[h(x) \neq y\right]$$

EXPECTED RISK

Deterministic	Randomized
$\mathcal{R}(h) = \mathbb{E}_{x,y}[\ell(h,x,y)]$	$\mathcal{R}(\mathbf{h}_\mu) = \mathbb{E}_{x,y}\mathbb{E}_{h\sim\mu}[\ell(h,x,y)]$
$\mathcal{R}_\epsilon(h) = \mathbb{E}_{x,y}[\ \sup_{x'\in B_\epsilon(x)}\ \ell(h,x',y)]$	$\mathcal{R}_\epsilon(\mathbf{h}_\mu) = \mathbb{E}_{x,y}\left[\ \sup_{x'\in B_\epsilon(x)}\ \mathbb{E}_{h\sim\mu}[\ell(h,x',y)]\right]$

We will focus on the case of the 0-1 loss

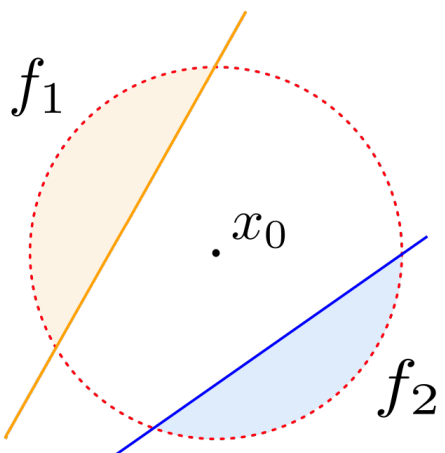
$$\ell^{0\text{-}1}(h,x,y) = \mathbb{1}\left[h(x) \neq y\right]$$

We can ask ourselves:

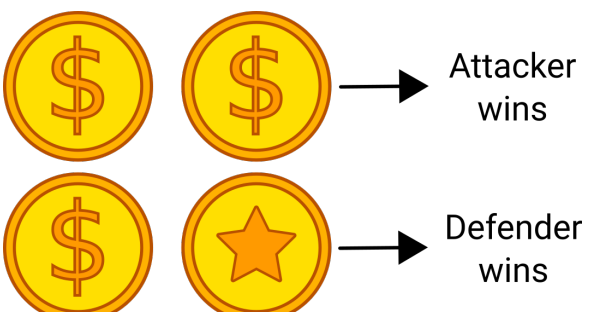
- How to attack $\mathbb{E}_{h\sim\mu}[\ell(h,x,y)]$???
- Do we get something better if we randomize?

Toy example

Suppose we have two classifiers available, and focus on the point (x_0, y_0).
What can we say about the risk of the mixture of f_1, f_2 ?

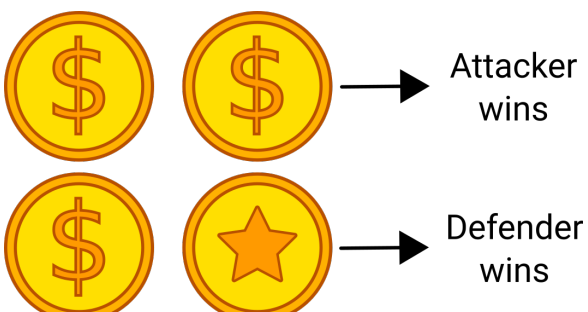


Matching pennies game



The *Matching pennies* game is a simple example with only **mixed** Nash equilibrium.

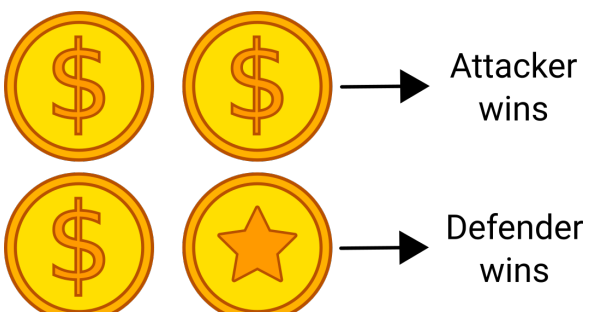
Matching pennies game



The *Matching pennies* game is a simple example with only **mixed** Nash equilibrium.

Instead of choosing an action, choose a distribution over actions *i.e. Toss the coin instead of choosing a side.*

Matching pennies game

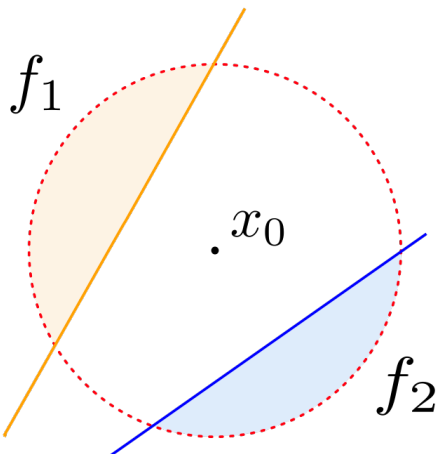


The *Matching pennies* game is a simple example with only **mixed** Nash equilibrium.

Instead of choosing an action, choose a distribution over actions *i.e. Toss the coin instead of choosing a side.*

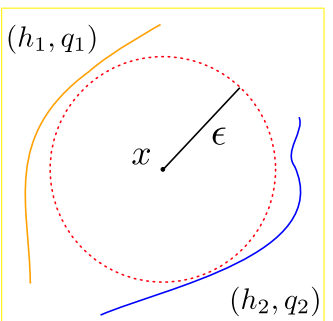
The *matching pennies* game shows mixed strategies can be strictly better.

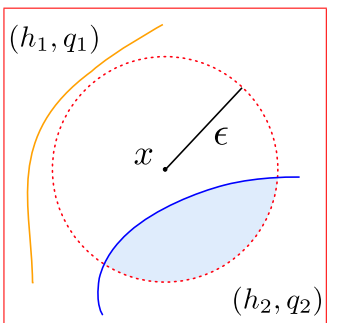
Toy example (cont)

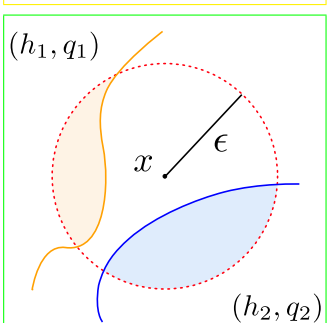


Mixing classifiers that are **vulnerable** but not **simultaneously vulnerable** creates a situation reminiscent of the game of *matching pennies*.

But ...


(h_1, q_1)
x
ϵ
(h_2, q_2)


(h_1, q_1)
x
ϵ
(h_2, q_2)


(h_1, q_1)
x
ϵ
(h_2, q_2)

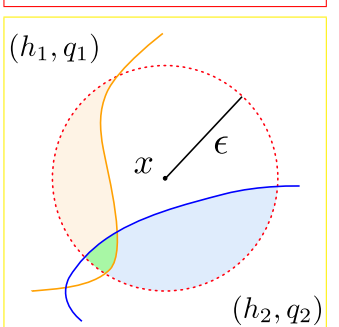

(h_1, q_1)
x
ϵ
(h_2, q_2)

Table of Contents

Randomization can improve robustness: The Matching Penny Gap

Note that by Jensen's inequality,

$$\mathcal{R}_\epsilon(\mathbf{h}_\mu) \leq \mathbb{E}_{h\sim\mu}\left[\mathcal{R}_\epsilon(h)\right]$$

Randomization can improve robustness: The Matching Penny Gap

Note that by Jensen's inequality,

$$\mathcal{R}_\epsilon(\mathbf{h}_\mu) \leq \mathbb{E}_{h\sim\mu}\left[\mathcal{R}_\epsilon(h)\right]$$

In other words, the adversarial risk of a mixture of classifiers is **at most** the average adversarial risk.

Randomization can improve robustness: The Matching Penny Gap

Note that by Jensen's inequality,

$$\mathcal{R}_\epsilon(\mathbf{h}_\mu) \leq \mathbb{E}_{h\sim\mu}\left[\mathcal{R}_\epsilon(h)\right]$$

In other words, the adversarial risk of a mixture of classifiers is **at most** the average adversarial risk.

Can we do better than the best classifier $h \in \mathcal{H}$?

Randomization can improve robustness: The Matching Penny Gap

Definition 4.1 (Matching penny gap)

The matching penny gap of $\mathbf{h}_\mu$ *at* (x, y) *is:*

$$\pi_{\mathbf{h}_\mu}(x, y) = \underbrace{\mu(\mathcal{H}_{vb}(x, y))}_{\textit{ind. vul}} - \underbrace{\mu^{\max}(x, y)}_{\textit{simult. vul}}$$

where

$$\begin{aligned}
\mathcal{H}_{vb}(x, y) &= \{h \in \mathcal{H}_b : \exists x'_h \in B_\epsilon(x) \text{ such that } h(x'_h) \neq y\}, && \textit{individually vulnerable} \\
\mathfrak{H}_{svb}(x, y) &= \{\mathcal{H}' \subseteq \mathcal{H}_b : \exists x' \in B_\epsilon(x) \text{ such that } \forall h \in \mathcal{H}', h(x') \neq y\}, && \textit{families of sim. vulnerable} \\
\mu^{\max}(x, y) &= \sup_{\mathcal{H}' \in \mathfrak{H}_{svb}(x,y)} \mu(\mathcal{H}'). && \textit{max simultaneously vulnerable}
\end{aligned}$$

If $\pi_{\mathbf{h}_\mu}(x, y) > 0$*, we say that* $\mathbf{h}_\mu$ *is in* matching penny configuration *at* (x, y).

Toy example

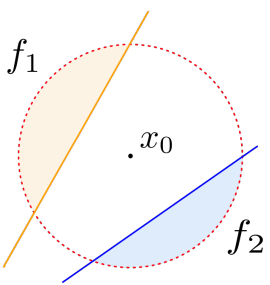


Figure. Let us $\pi_{\mathbf{h}_\mu}$ at the point (x_0, y) for this toy example. Both f_1, f_2 correctly predict the class y for x_0 in the white area, but they are fooled in the orange and blue areas, respectively.

$$\mathcal{H}_b = \{f_1, f_2\}, \qquad \mu = \left(\tfrac{1}{2}, \tfrac{1}{2}\right)$$

$$\mathcal{H}_{vb}(x_0, y) = \{f_1, f_2\} \implies \mu(\mathcal{H}_{vb}(x_0, y)) = 1$$

$$\mathfrak{H}_{svb}(x_0, y) = \{\{f_1\}, \{f_2\}\} \implies \mu^{\max}(x_0, y) = \tfrac{1}{2}$$

$$\therefore \quad \pi_{\mathbf{h}_\mu}(x_0, y) = 1 - \tfrac{1}{2} = \tfrac{1}{2}$$

Two vulnerable classifiers can be mixed to obtain $\frac{1}{2}$ expected adversarial risk !

Main result

Theorem 1

For a probabilistic classifier $\mathbf{h}_\mu : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ *constructed from a BHS* $\mathcal{H}_b$ *using any* $\mu \in \mathcal{P}(\mathcal{H}_b)$*,*

$$\mathcal{R}_\epsilon(\mathbf{h}_\mu) = \mathbb{E}_{h\sim\mu}\left[\mathcal{R}_\epsilon(h)\right] - \mathbb{E}_{(x,y)\sim\rho}[\pi_{\mathbf{h}_\mu}(x,y)]. \tag{1}$$

This theorem shows the link between the risk of a mixture $\mathbf{h}_\mu$ and the average risk. The gap is exactly the expected *matching penny gap*.

When does randomization improve robustness

Corollary 1

For $\mu \in \mathcal{P}(\mathcal{H}_b)$, $\mathcal{R}_\epsilon(\mathbf{h}_\mu) < \inf_{h \in \mathcal{H}_b} \mathcal{R}_\epsilon(h)$ if and only if the following condition holds.

$$\mathbb{E}_{(x,y)\sim\rho}[\pi_{\mathbf{h}_\mu}(x,y)] > \mathbb{E}_{h\sim\mu}[\mathcal{R}_\epsilon(h)] - \inf_{h\in\mathcal{H}_b} \mathcal{R}_\epsilon(h)$$

When does randomization improve robustness

Corollary 1

For $\mu \in \mathcal{P}(\mathcal{H}_b)$, $\mathcal{R}_\epsilon(\mathbf{h}_\mu) < \inf_{h \in \mathcal{H}_b} \mathcal{R}_\epsilon(h)$ if and only if the following condition holds.

$$\boxed{\mathbb{E}_{(x,y)\sim\rho}[\pi_{\mathbf{h}_\mu}(x,y)]} > \mathbb{E}_{h\sim\mu}[\mathcal{R}_\epsilon(h)] - \inf_{h\in\mathcal{H}_b} \mathcal{R}_\epsilon(h)$$

- Randomized classifiers are better if their expected matching penny gap is high.

When does randomization improve robustness

Corollary 1

For $\mu \in \mathcal{P}(\mathcal{H}_b)$, $\mathcal{R}_\epsilon(\mathbf{h}_\mu) < \inf_{h \in \mathcal{H}_b} \mathcal{R}_\epsilon(h)$ if and only if the following condition holds.

$$\mathbb{E}_{(x,y)\sim\rho}[\pi_{\mathbf{h}_\mu}(x,y)] > \boxed{\mathbb{E}_{h\sim\mu}[\mathcal{R}_\epsilon(h)] - \inf_{h\in\mathcal{H}_b} \mathcal{R}_\epsilon(h)}$$

- Randomized classifiers are better if their expected matching penny gap is high.
- RHS tells us that the individual $h \in \mathcal{H}_b$ should have similar robustness.

Table of Contents

Deterministic ensembles

Ensembles have been used widely for all learning tasks.
Intuitively, diversity has been considered a key component for performance [KW03; Kun14].

Deterministic ensembles

Ensembles have been used widely for all learning tasks.
Intuitively, diversity has been considered a key component for performance [KW03; Kun14].

In the adversarial attacks literature, there has also been a few efforts to build **diverse ensembles** as a robust alternative.

COMBINING CLASSIFIERS

Deterministic ensembles and mixtures of classifiers are similar, but **not equal**.
Here, $f : \mathcal{X} \to \Delta^K$ (probits), and $\tilde{f}(x)$ denotes $\arg\max_k f(x) \in \{e_1, \ldots, e_K\}$.

[1] Here we are comparing with the combining methods used in the ensemble methods we are going to see next.

COMBINING CLASSIFIERS

Deterministic ensembles and mixtures of classifiers are similar, but **not equal**.
Here, $f : \mathcal{X} \to \Delta^K$ (probits), and $\tilde{f}(x)$ denotes $\arg\max_k f(x) \in \{e_1, \ldots, e_K\}$.

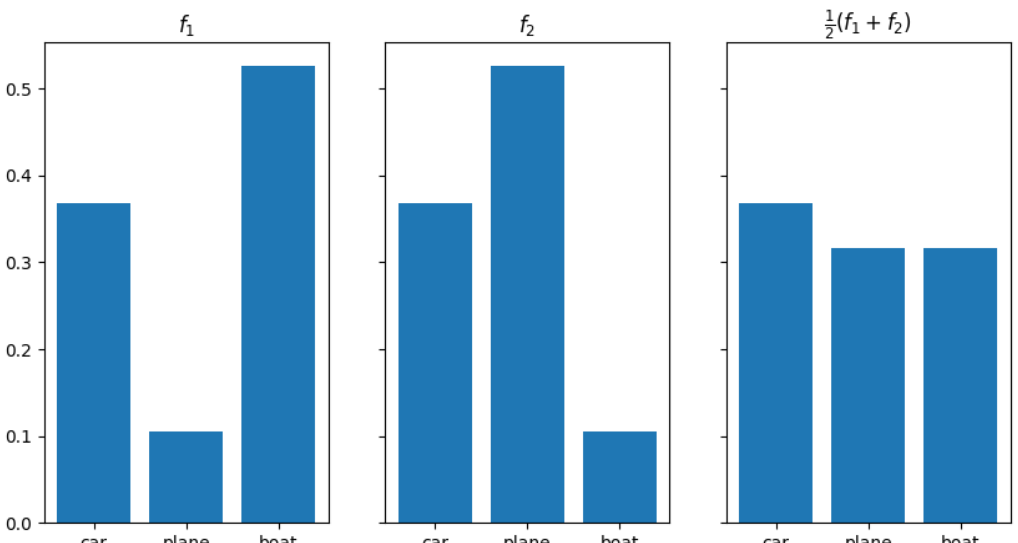
Ensembles[1]	Mixtures
$f(x) = \arg\max_k \left(\sum_i w_i f_i(x)\right)_k$	$\sum_i w_i \tilde{f}_i(x)$
$\ell^{0\text{-}1}(f, x, y) = \mathbb{1}\left[\arg\max_k \left(\sum_i w_i f_i(x)\right)_k \neq y\right]$	$\ell^{0\text{-}1}(f, x, y) = 1 - \left(\sum_i w_i \tilde{f}_i(x)\right)_y$

What is the attacker trying to optimize in each case ?

[1] Here we are comparing with the combining methods used in the ensemble methods we are going to see next.

Combining classifiers - Toy example

Classes are $[car, plane, boat]$. Correct class is *car*



GAL [KQ19]

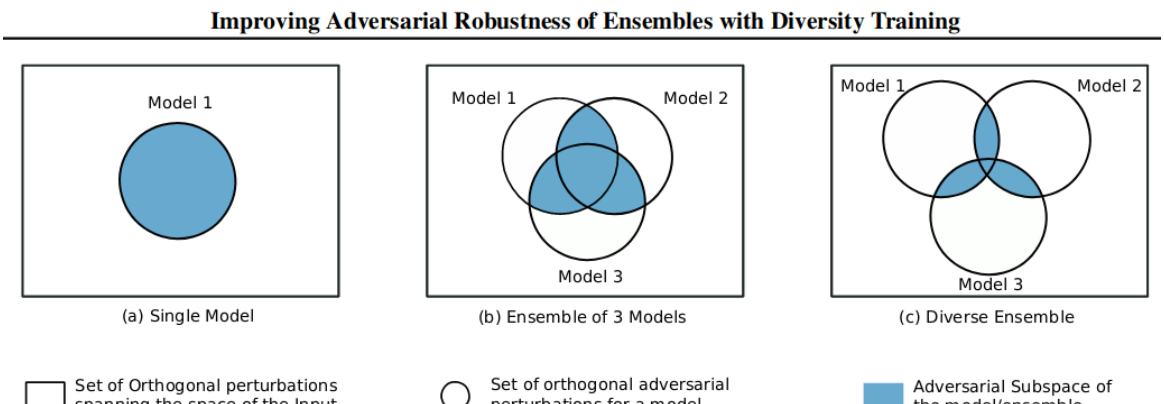
GAL, for *Gradient Alignment Loss*, minimizes the *coherence* of the gradients of the models.

$$\mathit{coherence}(\{\nabla_x J_i\}_{i=1}^m) = \max_{a \neq b} CS(\nabla_x J_a, \nabla_x J_b)$$

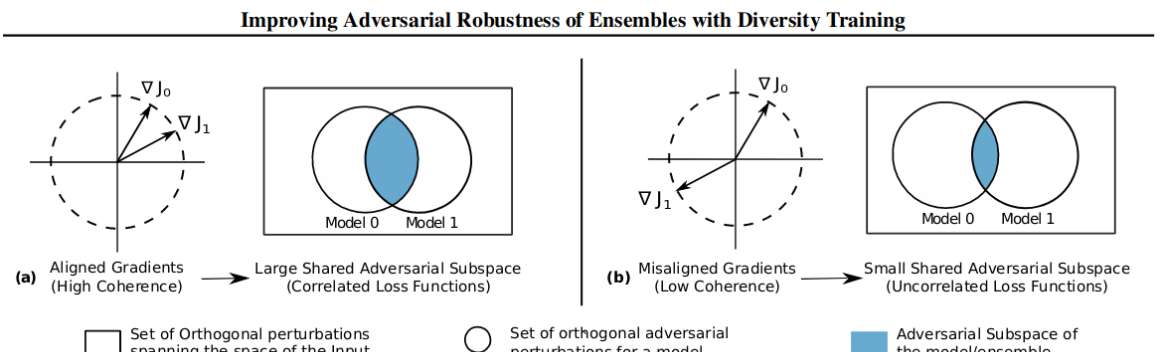
To do so, it minimizes a proxy

$$GAL = \log\left(\sum_{a<b} \exp(CS(\nabla_x J_a, \nabla_x J_b))\right)$$

GAL [KQ19]


Improving Adversarial Robustness of Ensembles with Diversity Training
Model 1
(a) Single Model
Model 1
Model 2
Model 3
(b) Ensemble of 3 Models
Model 1
Model 2
Model 3
(c) Diverse Ensemble
Set of Orthogonal perturbations spanning the space of the Input
Set of orthogonal adversarial perturbations for a model
Adversarial Subspace of the model/ensemble

GAL [KQ19]


Improving Adversarial Robustness of Ensembles with Diversity Training
∇J_0
∇J_1
Model 0
Model 1
(a)
Aligned Gradients
(High Coherence)
Large Shared Adversarial Subspace
(Correlated Loss Functions)
∇J_0
∇J_1
Model 0
Model 1
(b)
Misaligned Gradients
(Low Coherence)
Small Shared Adversarial Subspace
(Uncorrelated Loss Functions)
Set of Orthogonal perturbations
spanning the space of the Input
Set of orthogonal adversarial
perturbations for a model
Adversarial Subspace of
the model/ensemble

ADP [PAN+19]

ADP, for *Adaptive Diversity Promoting regularizer*, operates on the probits and not on gradients.

$$ADP_{\alpha,\beta}(x, y) = \alpha \cdot \underbrace{\mathcal{H}(\mathcal{F})}_{entropy} + \beta \cdot \underbrace{\log(\mathbb{ED})}_{alignment}$$

ADP [Pan+19]

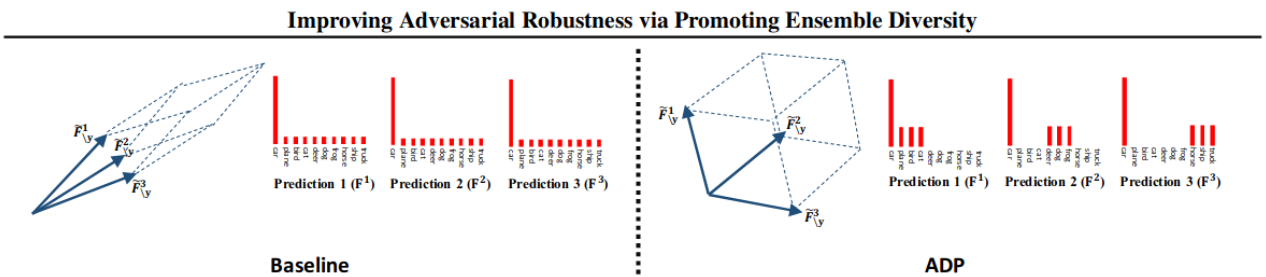


Figure 1. Illustration of the ensemble diversity. **Baseline:** Individually training each member of the ensemble. **ADP:** Simultaneously training all the members of the ensemble with the ADP regularizer. The left part of each panel is the normalized non-maximal predictions.

Other methods

- GPMR [Dab+20]
- DVERGE [Yan+20]
- TRS [Yan+21]

References I

[ACW18] Anish Athalye, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples". In: *International conference on machine learning*. PMLR. 2018, pp. 274–283.

[Dab+20] Ali Dabouei et al. "Exploiting joint robustness to adversarial perturbations". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1122–1131.

[Dhi+18] Guneet S. Dhillon et al. "Stochastic Activation Pruning for Robust Adversarial Defense". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL: https://openreview.net/forum?id=H1uR4GZRZ.

[DS22] Hassan Dbouk and Naresh Shanbhag. "Adversarial Vulnerability of Randomized Ensembles". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 4890–4917.

References II

[HRF19] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. "Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 588–597.

[KQ19] Sanjay Kariyappa and Moinuddin K. Qureshi. "Improving Adversarial Robustness of Ensembles with Diversity Training". In: *CoRR* abs/1901.09981 (2019). arXiv: 1901.09981. URL: http://arxiv.org/abs/1901.09981.

[Kun14] Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2014.

[KW03] Ludmila I Kuncheva and Christopher J Whitaker. "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy". In: *Machine learning* 51 (2003), pp. 181–207.

[Liu+18] Xuanqing Liu et al. "Towards robust neural networks via random self-ensemble". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 369–385.

References III

[Pan+19] Tianyu Pang et al. "Improving adversarial robustness via promoting ensemble diversity". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 4970–4979.

[Pan+21] Konstantinos Panousis et al. "Local competition and stochasticity for adversarial robustness in deep learning". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 3862–3870.

[PCT21] Konstantinos P Panousis, Sotirios Chatzis, and Sergios Theodoridis. "Stochastic local winner-takes-all networks enable profound adversarial robustness". In: *arXiv preprint arXiv:2112.02671* (2021).

[Pin+19] Rafael Pinot et al. "Theoretical evidence for adversarial robustness through randomization". In: *Advances in Neural Information Processing Systems* 32 (2019).

[Pin+20] Rafael Pinot et al. "Randomization matters how to defend against strong adversarial attacks". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 7717–7727.

[Tra+20] Florian Tramer et al. "On adaptive attacks to adversarial example defenses". In: *Advances in neural information processing systems* 33 (2020), pp. 1633–1645.

References IV

[Wic+21] Matthew Wicker et al. "Bayesian inference with certifiable adversarial robustness". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 2431–2439.

[Xie+17] Cihang Xie et al. "Mitigating adversarial effects through randomization". In: *arXiv preprint arXiv:1711.01991* (2017).

[Yan+20] Huanrui Yang et al. "DVERGE: diversifying vulnerabilities for enhanced robust generation of ensembles". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 5505–5515.

[Yan+21] Zhuolin Yang et al. "Trs: Transferability reduced ensemble via promoting gradient diversity and model smoothness". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 17642–17655.

[Yu+21] Tianyuan Yu et al. "Simple and effective stochastic neural networks". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2021, pp. 3252–3260.