ICLR

Understanding the **Robustness** of **Randomized Feature Defense** Against **Query-Based Adversarial Attacks**

Quang H. Nguyen[1], Yingjie Lao[2], Tung Pham[3], Kok-Seng Wong[1], Khoa D. Doan[1]

Tufts UNIVERSITY | School of Engineering 2

VINUNIVERSITY College of Engineering and Computer Science 1

VinAI RESEARCH 3

Machine Learning as a Service (MLaaS)

Several ML developments have emerged with extensive applications.

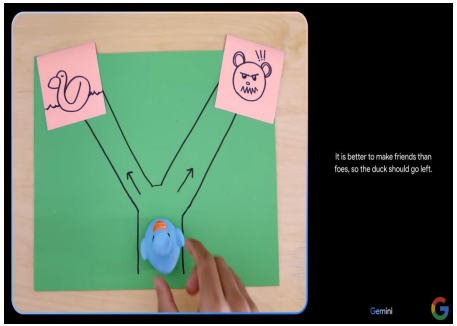
AI companies **expose** their ML assets (or models) for **external access**

and **users** can **query** these ML models for AI needs!

You
write a paper for iclr conference

ChatGPT
Title: Leveraging Deep Reinforcement Learning for Adaptive Curriculum Generation in Education

Abstract:
Deep Reinforcement Learning (DRL) has shown remarkable success in various domains, including gaming, robotics, and recommendation systems. In this paper, we explore the potential of applying DRL techniques to the domain of education, specifically focusing on adaptive curriculum generation. Traditional educational systems often follow a one-size-fits-all approach, where the same curriculum is delivered to all students regardless of their individual learning pace, preferences, and abilities. However, this approach may not be optimal for maximizing student engagement and learning outcomes.

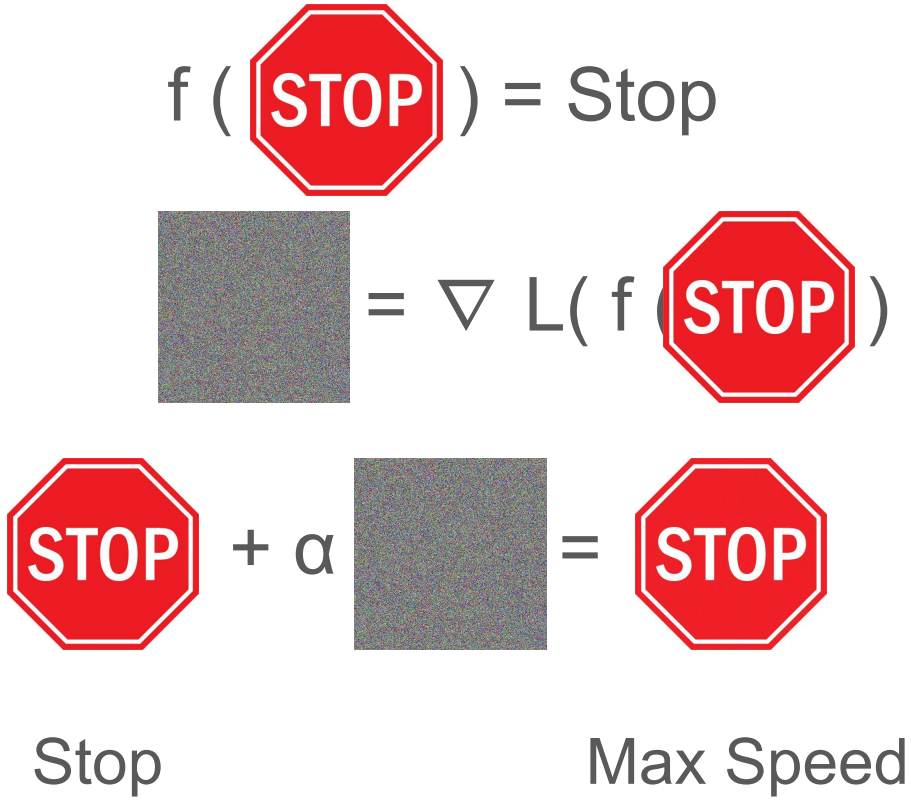


Adversarial attacks

Deep neural networks (**DNNs**) are **vulnerable** to **imperceptible noise**.

Attackers can employ **adversarial attack** to change the prediction of the model by **slightly perturbing the input**.

This raises a **serious threat** in MLaaS.



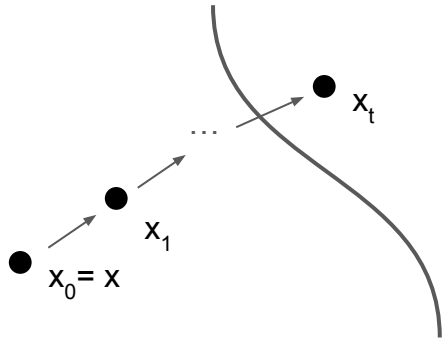
Adversarial attacks with more constraints

White-box attacks require access to the weight of the model to compute gradient.

Black-box attacks find adversarial samples by iteratively querying the model.

$$\hat{\nabla} L \approx \frac{L(f(x+u)) - L(f(x))}{\|u\|_2} u$$

This is threatening to **any** ML service.



Existing defensive methods

Adversarial Training

$$\min_{\theta} E\left[\max_{||\delta||\leq\epsilon} L(f(x+\delta),\, y)\right]$$

- Computationally expensive
- Clean-accuracy significantly degrades

Randomized Smoothing

$$g(x) = \arg\max_c P[f(x+\epsilon) = c],\ \epsilon \sim N\left(0, \sigma^2 I\right)$$

- Learn the noise (aka, access to training phase)
- Ensembling during inference (aka, high inference cost)

Users don't want to **sacrifice better performance** and **low cost** of unprotected models

Deceiving the attacker

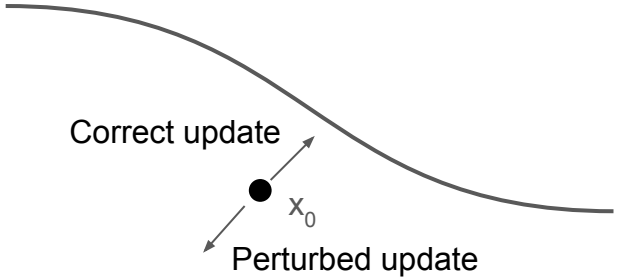
Random noise defense: Slightly adjusting the model output (with trivial impact on its accuracy) can lead the attacker in the wrong direction.

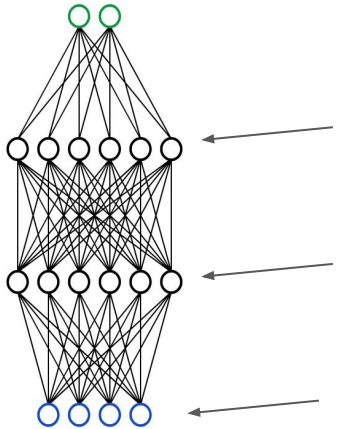
Why **Random noise defense:**

- It is *plug-and-play*
- It is *lightweight*
- It *doesn't* require *training*
- It allows control of *robustness-accuracy trade-off*

But where should we insert noise?

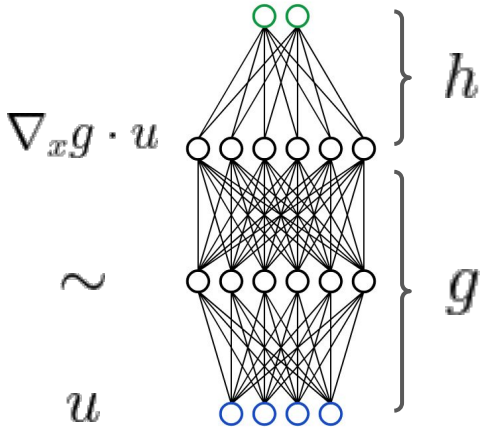
$$L(f'(x+u)) - L(f'(x)) \neq L(f(x+u)) - L(f(x))$$





Adding noise to input or internal features?

Adding a vector u to input has similar effect to adding $\nabla_x g \cdot u$ to the output of g.



Theorem 1. *Assuming the proposed random vector u is sampled from a Gaussian $\mathcal{N}(0, \mu I)$, the model is decomposed into $f = g \circ h$, and the defense adds a random noise $\delta \sim \mathcal{N}(0, \nu I)$ to the output of h. At input x, the probability that the attacker chooses an opposite action positively correlates with*

$$\arctan\left(-\left(\frac{2\nu}{\mu}\frac{\|\nabla_{h(x)}(\mathcal{L}\circ g)\|_2^2}{\|\nabla_x(\mathcal{L}\circ f)\|_2^2}\right)^{-0.5}\right).$$

- \nu is noise added by the defender
- \mu is perturbation radius of the attack

Depends on the property of the model!

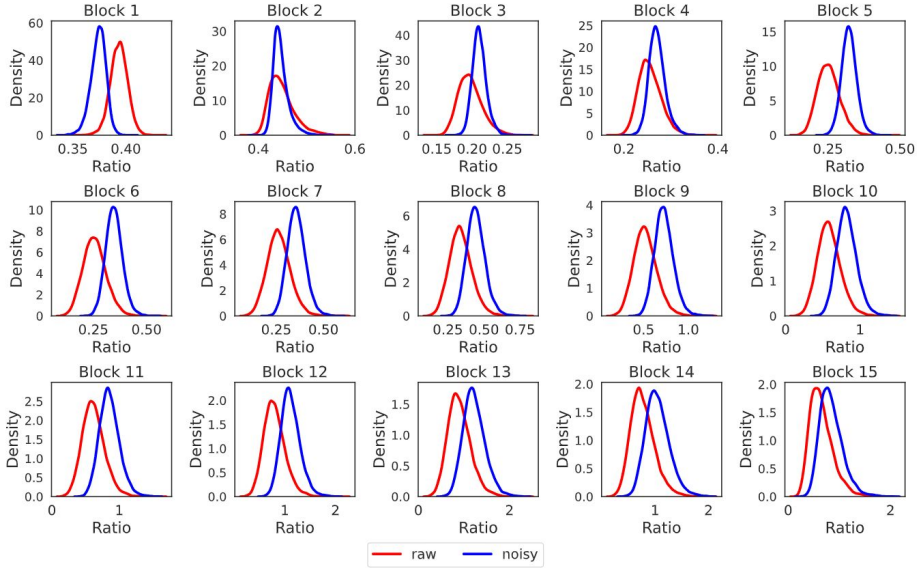
Higher $\frac{\|\nabla_{h(x)}(\mathcal{L}\circ g)\|_2^2}{\|\nabla_x(\mathcal{L}\circ f)\|_2^2}$

= Higher chance to fool the attack

Robustness of Randomizing a Layer

- Higher ratios for input closer to decision boundary **(blue)**

⇨ the probability that our defense fools the adversary **increases** during the attack.



The **gradient-norm ratio** of **test input** (red) & **input close to decision boundary** (blue) at different randomized layers

Robustness against Query-based Attacks

Randomized noise scale is selected at 1% or 2% accuracy decrease (using a small test set)

Randomizing the features shows **better robustness** than randomizing the input

- **across different models** (including CNNs and ViTs)
- **against various attacks** (gradient-based and random search)
- **similar robustness are observed** on other datasets (e.g., CIFAR10)

Randomly search for perturbation

Approximate the gradient

Randomly search for perturbation

Model	Method	Acc	Square		NES		SignHunt	
			1000	10000	1000	10000	1000	10000
ResNet50	Base	80.37	3.5	0.2	36.2	4.3	6.6	0.4
	Input	79.18 (≈ 1%)	40.3	39.5	63.8	23.9	47.6	45.4
		78.46 (≈ 2%)	41.1	39.8	**69.4**	**41.5**	49.3	47.2
	Feature	79.70 (≈ 1%)	37.0	36.0	56.7	16.8	46.3	43.4
		78.43 (≈ 2%)	**42.0**	**41.5**	65.6	40.6	**51.3**	**49.3**
VGG19	Base	74.21	0.1	0.0	19.6	0.0	0.4	0.0
	Input	73.24 (≈ 1%)	7.7	6.9	32.1	1.5	18.3	17.0
		71.43 (≈ 2%)	18.7	17.8	47.4	11.5	28.3	27.1
	Feature	72.66 (≈ 1%)	22.4	21.6	50.1	18.5	34.6	**32.9**
		71.21 (≈ 2%)	**23.3**	**22.2**	**55.1**	**28.4**	**36.5**	32.8
DeiT	Base	82.00	6.4	0.0	46.7	0.8	22.3	0.0
	Input	80.10 (≈ 1%)	67.7	67.2	**75.8**	65.9	64.4	63.6
		79.60 (≈ 2%)	66.6	66.0	75.7	**67.1**	64.9	64.3
	Feature	80.80 (≈ 1%)	**69.7**	**69.1**	75.0	59.1	**66.4**	64.1
		79.76 (≈ 2%)	69.3	69.0	75.1	65.3	66	**64.3**
ViT	Base	79.15	5.7	0.0	45.7	7.3	5.1	0.0
	Input	78.28 (≈ 1%)	58.8	58.1	70.8	51.4	53.1	52.2
		77.09 (≈ 2%)	61.3	60.9	70.6	**59.2**	53.7	52.7
	Feature	78.20 (≈ 1%)	60.6	60.2	69.1	47.5	54.0	52.9
		77.18 (≈ 2%)	**63.7**	**62.9**	**72.2**	58.1	**57.0**	**55.3**

Robustness on Imagenet

Robustness against Decision-based and Adaptive Attacks

Against **decision-based attacks** (that use hard labels)

Model	Method	Acc	RayS	SignFlip
ResNet50	Base	97.66	0.1	20.5
	AAA	97.70	0.1	20.4
	Input	93.52	12.0	**85.5**
	Feature	92.10	**14.4**	82.5
VGG19	Base	96.28	0.0	6.4
	AAA	96.30	0.1	5.7
	Input	93.42	8.1	**86.0**
	Feature	93.48	**15.4**	76.5

Randomizing the features achieves comparable robustness against decision-based attacks

Against **adaptive attacks** (search direction is averaged over M queries)

Attacks	Methods	VGG19						ResNet50					
		Acc	$M=1$	$M=5$		$M=10$		Acc	$M=1$	$M=5$		$M=10$	
			QC=1000	QC=1000	QC=5000	QC=1000	QC=10000		QC=1000	QC=1000	QC=5000	QC=1000	QC=10000
Square	Input	94.92	30.6	24.2	10.5	30.2	3.2	95.32	52.9	42.0	34.8	35.0	13.3
	Feature	94.93	61.0	53.0	45.5	46.7	23.1	95.21	54.5	45.1	40.4	37.3	21.1
NES	Input	94.92	89.5	93.4	82.1	94.4	78.8	95.32	92.4	94.0	91.3	93.9	90.7
	Feature	94.93	92.2	94.8	88.4	94.5	86.0	95.21	91.8	93.8	90.8	94.0	90.4
SignHunt	Input	94.92	22.7	15.9	10.4	23.3	7.6	95.32	29.9	17.6	13.5	21.1	9.4
	Feature	94.93	43.2	27.1	23.0	31.7	17.0	95.21	35.1	17.3	16.4	21.5	11.3

While **robustness decreases** as M increases

randomized feature defense still achieve strong robustness against some adaptive attacks

Conclusion

- We **propose a randomized feature defense** for black-box adversarial attacks.
- We provide **theoretical** and **empirical** analysis for the effect of adding noise.
- Our defense
  - is **plug-and-play**
  - is **lightweight**
  - **does not** require **training**
  - allows control of **robustness-accuracy trade-off**

- **This defense can be combined w. other defenses** for even better robustness (discussion in the paper, along with several other experiments)

THANK YOU!

Code: https://github.com/mail-research/randomized_defenses/
Contact: quanghngnguyen@gmail.com / khoadoan106@gmail.com
Lab: https://khoadoan.me

Robustness against Decision-based and Adaptive Attacks

Our defense is effective against decision-based and adaptive attacks

Table 5: Robustness against decision-based attacks (CIFAR10)

Model	Method	Acc	RayS	SignFlip
ResNet50	Base	97.66	0.1	20.5
	AAA	97.70	0.1	20.4
	Input	93.52	12.0	**85.5**
	Feature	92.10	**14.4**	82.5
VGG19	Base	96.28	0.0	6.4
	AAA	96.30	0.1	5.7
	Input	93.42	8.1	**86.0**
	Feature	93.48	**15.4**	76.5

Table 7: Defenses against adaptive attacks on CIFAR10

Attacks	Methods	VGG19						ResNet50					
		Acc	$M=1$	$M=5$		$M=10$		Acc	$M=1$	$M=5$		$M=10$	
			QC=1000	QC=1000	QC=5000	QC=1000	QC=10000		QC=1000	QC=1000	QC=5000	QC=1000	QC=10000
Square	Input	94.92	30.6	24.2	10.5	30.2	3.2	95.32	52.9	42.0	34.8	35.0	13.3
	Feature	94.93	61.0	53.0	45.5	46.7	23.1	95.21	54.5	45.1	40.4	37.3	21.1
NES	Input	94.92	89.5	93.4	82.1	94.4	78.8	95.32	92.4	94.0	91.3	93.9	90.7
	Feature	94.93	92.2	94.8	88.4	94.5	86.0	95.21	91.8	93.8	90.8	94.0	90.4
SignHunt	Input	94.92	22.7	15.9	10.4	23.3	7.6	95.32	29.9	17.6	13.5	21.1	9.4
	Feature	94.93	43.2	27.1	23.0	31.7	17.0	95.21	35.1	17.3	16.4	21.5	11.3

Performing the attack

White-box attacks require access to the weight of the model to compute gradient.

They cannot attack a system where only the output is available

Existing defensive methods

Adversarial training: *Train* the model with adversarial examples.

Randomized smoothing: Inject noise to the model and *ensemble* several predictions.

Random noise defense: Inject small noise to the input to mislead the attack.

Adversarial Attack on Attackers (AAA): Optimize output to fool *score-based attacks*.

Problems?

- Computationally expensive.
- Might decrease the performance on normal data.
- AAA does not work against decision-based attacks.

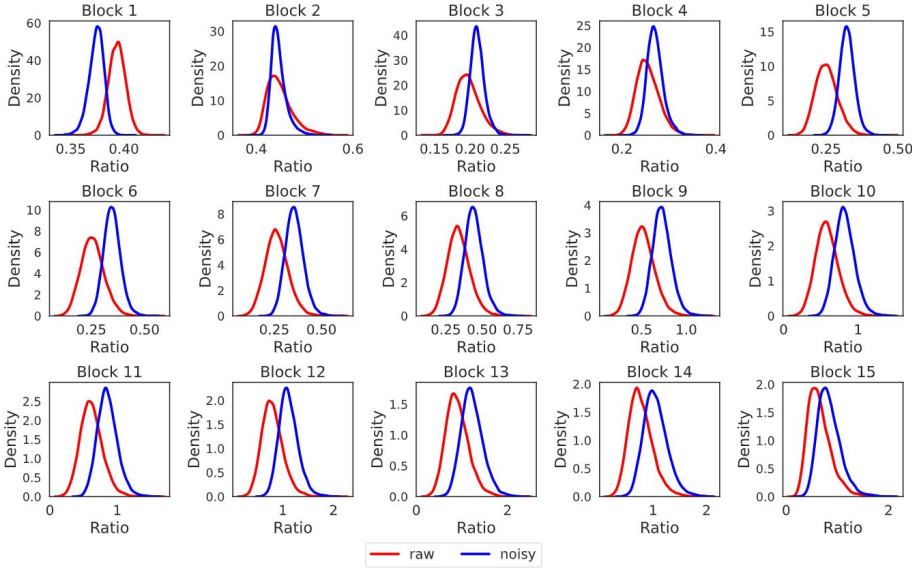
Robustness of Randomizing a Layer

Robustness is higher at input closer to decision boundary (h

The ratio of gradient norms is **higher** when the input is **closer** to the decision boundary.

The probability that our defense fools the adversary **increases** during the attack process.

Adding noise to hidden features is better in terms of defending.



The **gradient-norm ratio** of **test input** (red) & **input close to decision boundary**(blue) at different randomized layers