# Defending Against Multiple and Unforeseen Adversarial Videos

Shao-Yuan Lo, Vishal M. Patel

Johns Hopkins University

IEEE Transactions on Image Processing (TIP), 2021

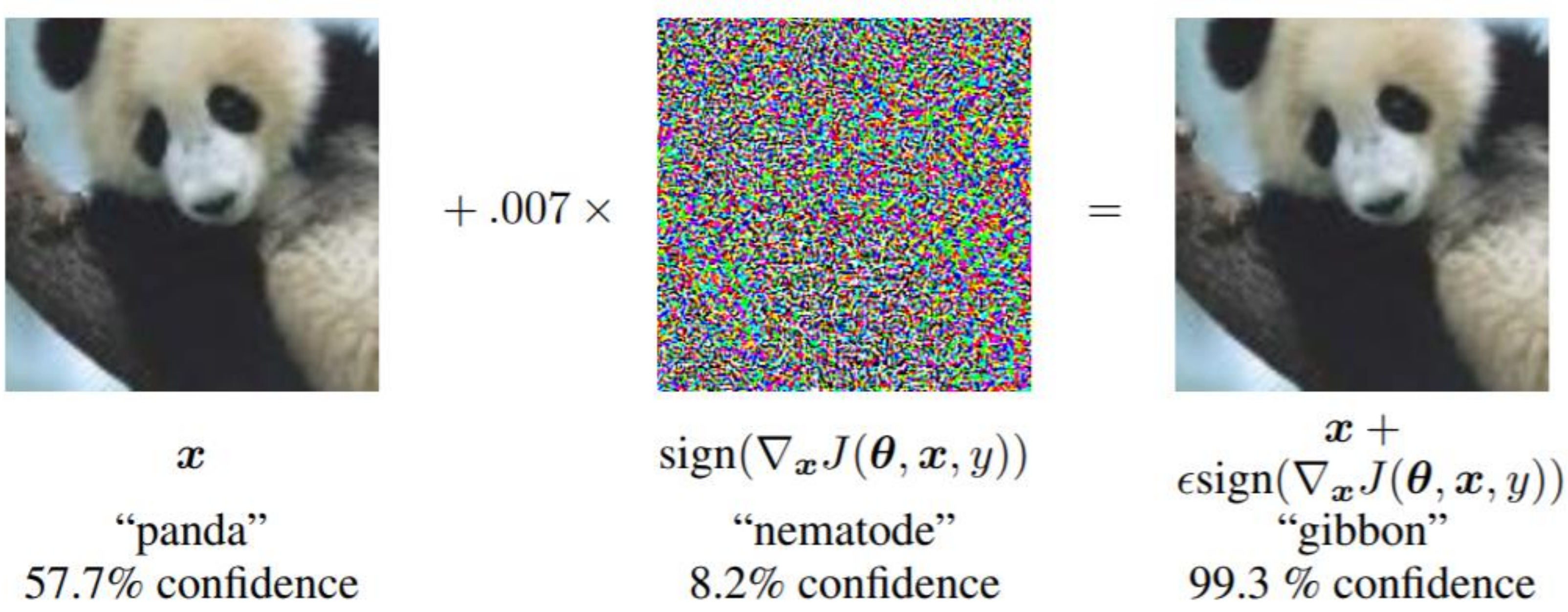
# What's Adversarial Example?

$$x_{adv} = x + \delta$$

$$f(\boldsymbol{x}_{adv}) \neq y$$

# What's Adversarial Example?

- Adversarial examples are visually **similar** to **human** but can **fool** well-trained **deep networks**.
- Deep networks are **vulnerable** to adversarial examples.




[Goodfellow et al. ICLR'15]

# Generate Adversarial Examples

- Train a model
  - **min Loss(f(x), y; θ)**
  - **Minimize** the loss function w.r.t. **model parameters θ**

- Generate adversarial examples
  - Most common method: Gradient-based method, e.g., FGSM.
  - **max Loss(f(x+δ), y; θ)**
  - **Maximize** the loss function w.r.t. **adversarial perturbation δ**

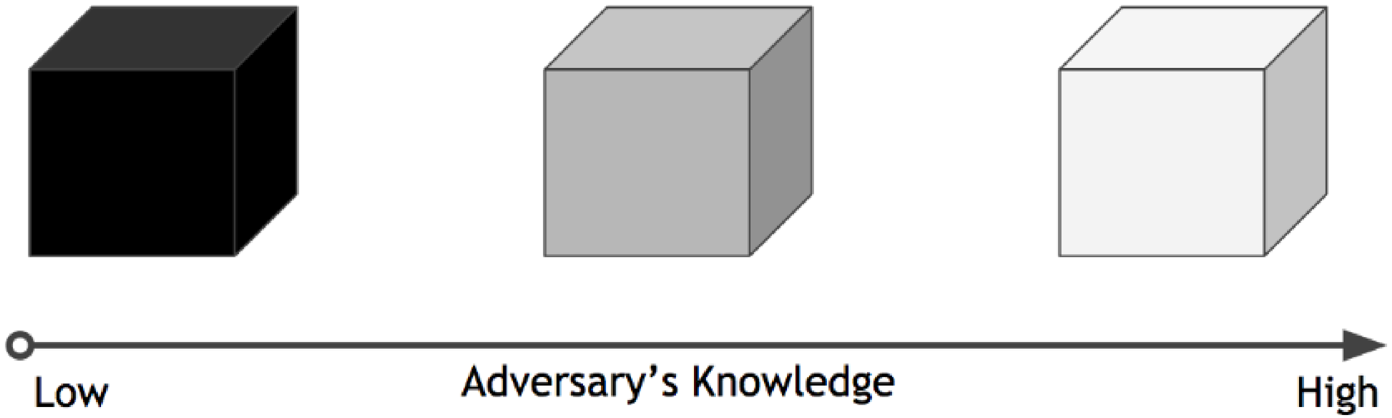
# Generate Adversarial Examples

- Generate adversarial examples
  - Most common method: Gradient-based method, e.g., FGSM.
  - **max Loss(f(x+δ), y; θ)**
  - **Maximize** the loss function w.r.t. **adversarial perturbation δ**

- Perturbation budget **‖δ‖**
  - Constrain the **magnitude** of perturbation, e.g., **Lp-norm**.
  - Constrain the **region** of perturbation, e.g., **patch attack**.

# Adversary's Knowledge

- White-box attack
- Black-box attack
- Gray-box attack




https://slidetodoc.com/unclassified-if-you-know-the-enemy-and-know

# Untargeted/Targeted Attacks

- Untargeted attack

$$f(\boldsymbol{x}_{adv}) \neq y$$

$$L_{adv}(\boldsymbol{x}) = -L(\boldsymbol{x}, y)$$

- Targeted attack

$$f(\boldsymbol{x}_{adv}) = y_{adv}, \quad y_{adv} \neq y$$

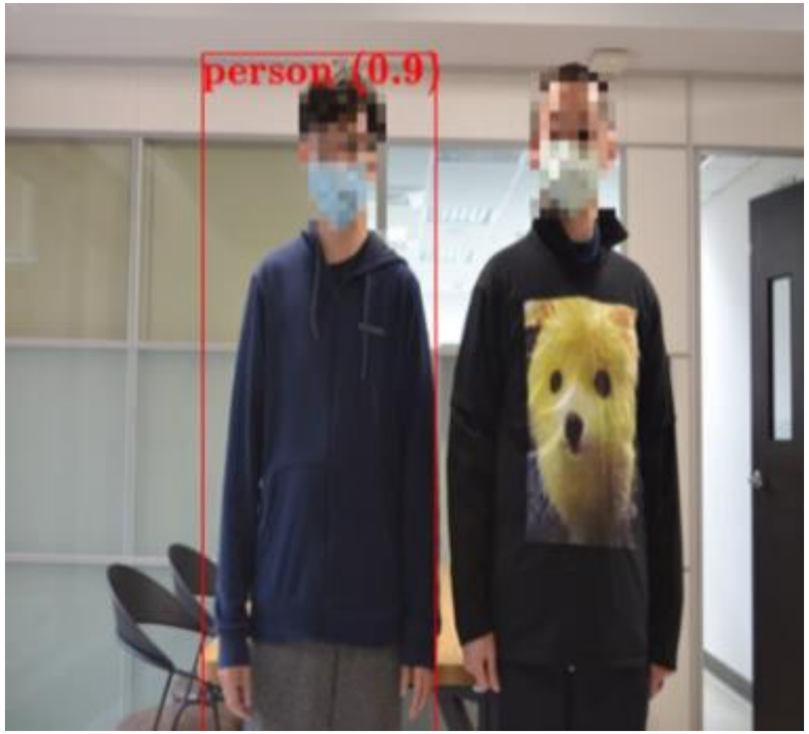
$$L_{adv}(\boldsymbol{x}) = L(\boldsymbol{x}, y_{adv})$$

# Adversarial Examples in Different Types

**Original** **Adversarial** **Prediction** **Original** **Adversarial** **Prediction**


[Wu et al. ICLR'20]

# Adversarial Examples in Physical World



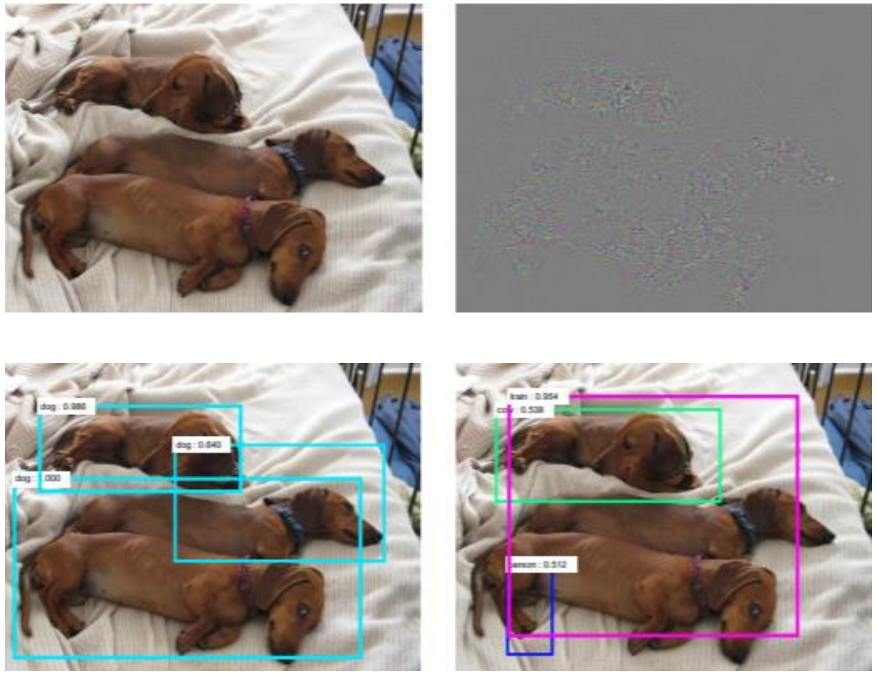

[Hu et al. ICCV'21]

[Ranjan et al. ICCV'19]

# Adversarial Examples in Different Tasks

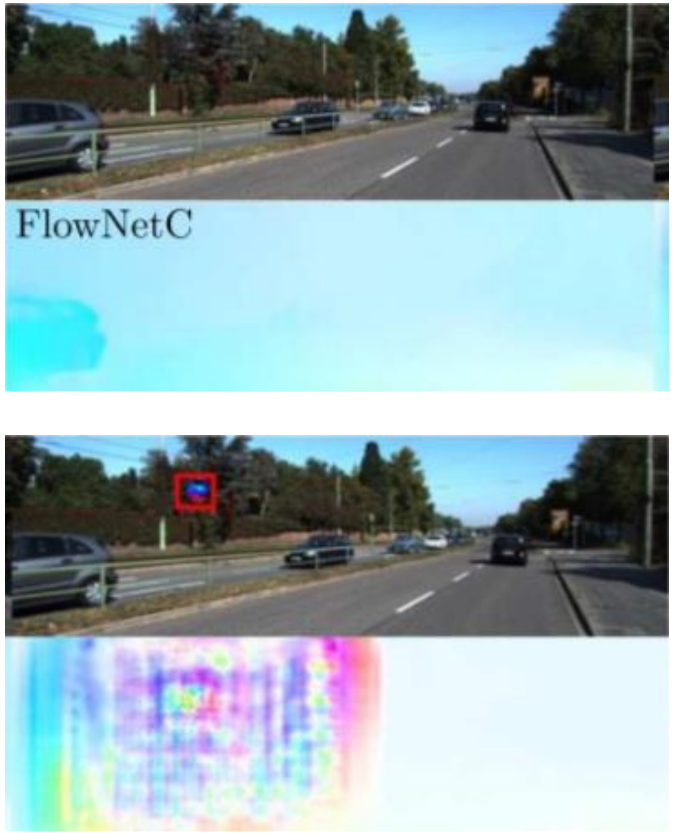
**Semantic segmentation**


**Object detection**



[Xie et al. ICCV'17]

**Optical flow**




[Ranjan et al. ICCV'19]

# Adversarial Defenses

- **Image transformation**: Remove perturbations from input images.

$$f(\boldsymbol{x}_{adv}) \neq y$$

$$f(\boldsymbol{T}(\boldsymbol{x}_{adv})) = y$$

- **Adversarial training**: Enhance the robustness of networks itself.

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim\mathbb{D}} \left[ \max_{\delta\in\mathbb{S}} L(x+\delta, y; \theta) \right]$$

# Image Transformation-based Defenses

- Image preprocessing methods:
  - **Color precision reduction** (pixel value quantization)
  - **JPEG compression** (frequency domain quantization)
  - **Denoising** (Gaussian blur, median, mean, bilateral, non-local means, etc.)
  - **Color space** (RGB, HSV, YUV, LAB, etc.)
  - **Contrast** (histogram equalization)
  - **Noise injection** (add noise on adversarial examples)
  - **FFT perturbation** (similar to JPEG)
  - **Swirl** (rotation)
  - **Resizing**
  - **Gray scale**

[Das et al. KDD’18]
[Xu et al. NDSS’18]
[Guo et al. ICLR’18]
[Raff et al. CVPR’19]

- Generative model methods:
  - **Defense-GAN**
    [Samangouei et al. ICLR’18]
  - **PixelDefend**
    [Song et al. ICLR’18]

# Image Transformation-based Defenses

- [Athalye et al. ICML'19] proposed **adaptive attacks**, which defeat most image transformation-based defenses.
- Strong white-box attacks are generated through **gradients**, e.g., FGSM and PGD attacks.
- Image transformation-based defenses mostly rely on **gradient masking**, which can be defeated by adaptive attacks.
- Three types of masked gradients:
  - Shattered gradients ← BPDA
  - Stochastic gradients ← EOT
  - Exploding & vanishing gradients ← BPDA or EOT or Both

| Defense | Dataset | Distance | Accuracy |
|---|---|---|---|
| Buckman et al. (2018) | CIFAR | 0.031 ($\ell_\infty$) | 0%* |
| Ma et al. (2018) | CIFAR | 0.031 ($\ell_\infty$) | 5% |
| Guo et al. (2018) | ImageNet | 0.005 ($\ell_2$) | 0%* |
| Dhillon et al. (2018) | CIFAR | 0.031 ($\ell_\infty$) | 0% |
| Xie et al. (2018) | ImageNet | 0.031 ($\ell_\infty$) | 0%* |
| Song et al. (2018) | CIFAR | 0.031 ($\ell_\infty$) | 9%* |
| Samangouei et al. (2018) | MNIST | 0.005 ($\ell_2$) | 55%** |
| Madry et al. (2018) | CIFAR | 0.031 ($\ell_\infty$) | 47% |
| Na et al. (2018) | CIFAR | 0.015 ($\ell_\infty$) | 15% |

# Adversarial Training

- Adversarial training is a strong defense against white-box attacks.
- **Core idea: Train with adversarial examples.**
- Adversarial training does **not** cause masked gradients.
- It has been widely used as a standard baseline defense.

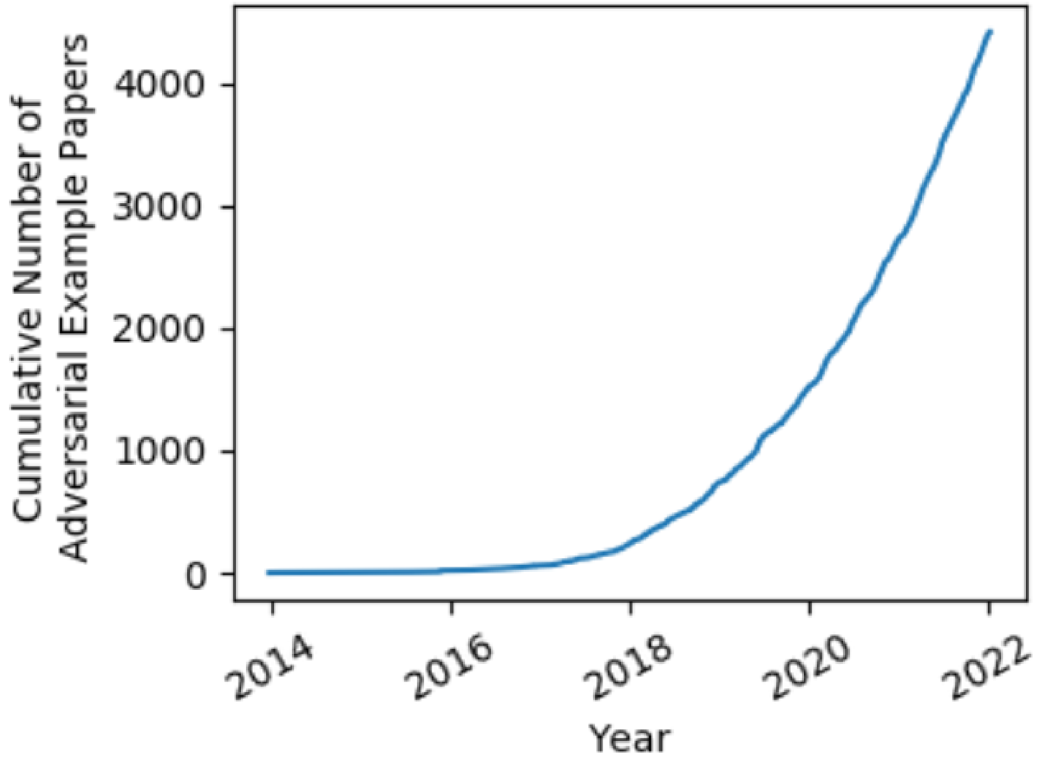
$$\underbrace{\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim\mathbb{D}} \left[ \underbrace{\max_{\delta\in\mathbb{S}} L(x+\delta, y; \theta)}_{\text{Generate adversarial examples}} \right]}_{\text{Train model parameters}}$$

[Madry et al. ICLR'18]

# Why Study Adversarial Examples?

- Deep learning models are being widely used in real-world applications, such as autonomous driving. Their **safety** is critical.

- We aim to build **robust** DL models that we can **trust**.




https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html

# Why Videos?

- Most research in adversarial examples focuses on static **images**.
- Adversarial attacks and defenses for **videos** are less explored.

- To the best of our knowledge, this work is the **first** defense against white-box attacks in the video domain.
- We provide **comprehensive baseline results** for adversarial robustness in the video domain.

# Adversarial Videos

- Video is a stack of consecutive images.
- A naïve way to generate adversarial videos:
  Use image-based method directly.

$$x^{adv} = x + \epsilon \cdot sign(\nabla_x L(x, y; \theta))$$

$$Image:\ x \in R^{C \times H \times W}$$

$$Video:\ x \in R^{\boldsymbol{F} \times C \times H \times W}$$

# Adversarial Framing (AF)

correct: Boston bull
unattacked: Boston bull
attacked: maypole

correct: ocarina
unattacked: loupe
attacked: maypole

correct: tusker
unattacked: tusker
attacked: maypole

correct: gas pump
unattacked: gas pump
attacked: maypole

correct: Egyptian cat
unattacked: tabby
attacked: maypole

Task: Action recognition
Dataset: UCF-101

| **Attack** | $W=1$ | $W=2$ | $W=3$ | $W=4$ |
|---|---|---|---|---|
| **None** | 85.95% | | | |
| **RF** | 82.57% | 80.53% | 81.11% | 79.74% |
| **BF** | 84.94% | 84.73% | 84.75% | 84.59% |
| **AF** | 65.77% | 22.12% | 9.45% | 2.05% |

[Zajac et al. AAAI'19]

# Salt-and-Pepper Attack (SPA)

- Add unbounded perturbations on a number of randomly selected pixels.
- The perturbation looks like salt-and-pepper noise.
- A kind of L0-norm attack.

- Decrease action recognition accuracy from **89.0%** to **8.4%** on UCF-101.

Clean SPA

# Adversarial Video Types

- PGD:
  Projective gradient descent
  [Madry et al. ICLR'18]
- ROA:
  Rectangular occlusion
  [Wu et al. ICLR'20]
- AF:
  Adversarial Framing
  [Zajac et al. AAAI'19]
- SPA:
  Salt-and-Pepper noise

# Adversarial Video Types

- PGD:
  Projective gradient descent
  [Madry et al. ICLR'18]

- ROA:
  Rectangular occlusion
  [Wu et al. ICLR'20]

- AF:
  Adversarial Framing
  [Zajac et al. AAAI'19]

- SPA:
  Salt-and-Pepper noise


**How to simultaneously defend against multiple types of attacks?**

# Problem: Multi-perturbation Robustness

- Standard adversarial training has poor **multi-perturbation robustness**.
- Training: $\delta_{PGD}$
- Test: Clean, $\delta_{PGD}$, $\delta_{ROA}$, $\delta_{AF}$, $\delta_{SPA}$

$$\underbrace{\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim\mathbb{D}} \Bigg[\underbrace{\max_{\delta\in\mathbb{S}} L(x+\delta, y; \theta)}_{\text{Generate one type of adversarial examples}}\Bigg]}_{\text{Train model parameters}}$$

Generate **one type** of adversarial examples

Train model parameters

[Madry et al. ICLR'18]

# Problem: Multi-perturbation Robustness

- Dataset: UCF-101 (action recognition)
- Model: 3D ResNeXt-101
- Attack setting:
  PGD Linf: ε=4/255, T=5
  ROA: patch size=30x30
  AF: width=10
  SPA: #pixels=100, T=5

| Model | Clean | PGD | ROA | AF | SPA | Mean | Union |
|---|---|---|---|---|---|---|---|
| No Defense | **89.0** | 3.3 | 0.5 | 1.6 | 8.4 | 20.6 | 0.0 |
| AT-PGD | 78.6 | **49.0** | 5.0 | 0.6 | 67.1 | 40.1 | 0.3 |
| AT-ROA | 82.6 | 12.5 | **69.0** | 54.0 | 17.6 | 47.1 | 7.9 |
| AT-AF | 84.6 | 7.1 | 3.9 | **80.5** | 12.2 | 37.7 | 2.1 |
| AT-SPA | 83.5 | 36.9 | 2.6 | 0.7 | **69.5** | 38.6 | 0.2 |

# Problem: Multi-perturbation Robustness

- **Average** adversarial training is better, but not enough.
- Training: Clean, $\delta_{PGD}$, $\delta_{ROA}$, $\delta_{AF}$, $\delta_{SPA}$
- Test: Clean, $\delta_{PGD}$, $\delta_{ROA}$, $\delta_{AF}$, $\delta_{SPA}$

$$\underbrace{\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim\mathbb{D}} \left[ \underbrace{\sum_{i=1}^{N} \max_{\delta_i \in \mathbb{S}_i} L(x+\delta_i, y; \theta)}_{\text{Generate multiple types of adversarial examples}} \right]}_{\text{Train model parameters}}$$

Generate **multiple types** of adversarial examples

Train model parameters

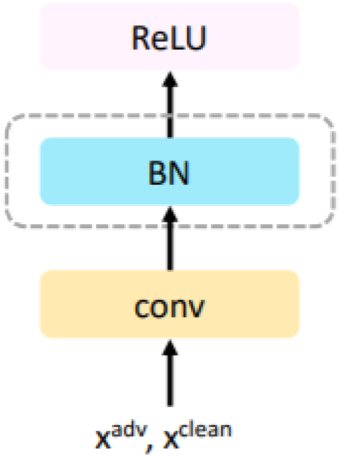
[Tramèr & Boneh NeurIPS'19]

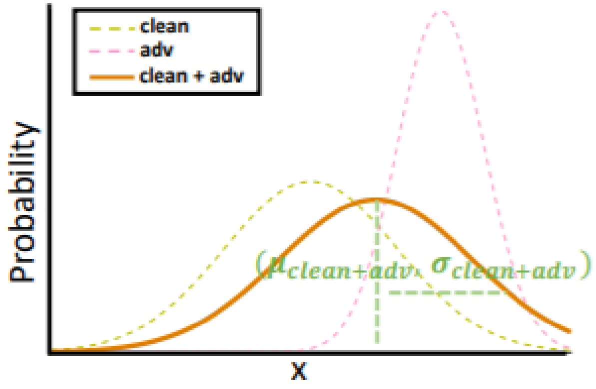
# Problem: Multi-perturbation Robustness

| Model | Clean | PGD | ROA | AF | SPA | Mean | Union |
|---|---|---|---|---|---|---|---|
| No Defense | **89.0** | 3.3 | 0.5 | 1.6 | 8.4 | 20.6 | 0.0 |
| AT-PGD | 78.6 | **49.0** | 5.0 | 0.6 | 67.1 | 40.1 | 0.3 |
| AT-ROA | 82.6 | 12.5 | **69.0** | 54.0 | 17.6 | 47.1 | 7.9 |
| AT-AF | 84.6 | 7.1 | 3.9 | **80.5** | 12.2 | 37.7 | 2.1 |
| AT-SPA | 83.5 | 36.9 | 2.6 | 0.7 | **69.5** | 38.6 | 0.2 |
| AVG [30] (NeurIPS'19) | 74.5 | 43.1 | 55.6 | 3.5 | 57.2 | 46.8 | 3.5 |

# Observation: Distinct Data Distributions

- Why average adversarial training is **not** an ideal strategy?
- Example: **Clean** vs. **PGD**.
- Clean and PGD have **distinct** data distributions.
- The statistics estimation at **BN** may be confused when facing a mixture distribution.

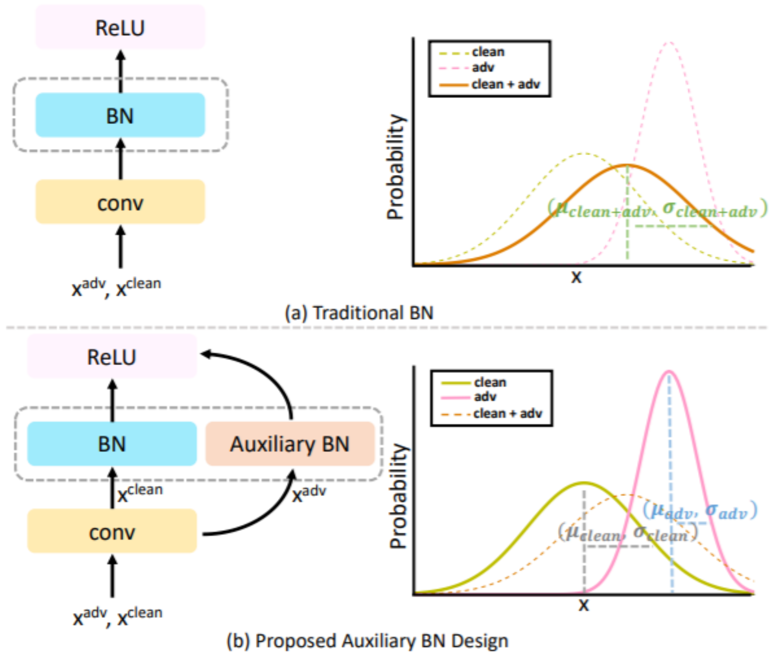



[Xie et al. CVPR'20]

# Observation: Distinct Data Distributions

- Example: **Clean** vs. **PGD**.
- An **auxiliary BN** guarantees that data from different distributions are normalized separately.




[Xie et al. CVPR'20]

# Extension for Multi-perturbation Robustness

- What about **multiple** attack types?
- Example: Clean, PGD, ROA, AF, SPA
- Our assumption: Different attack types have **distinct** data distributions.

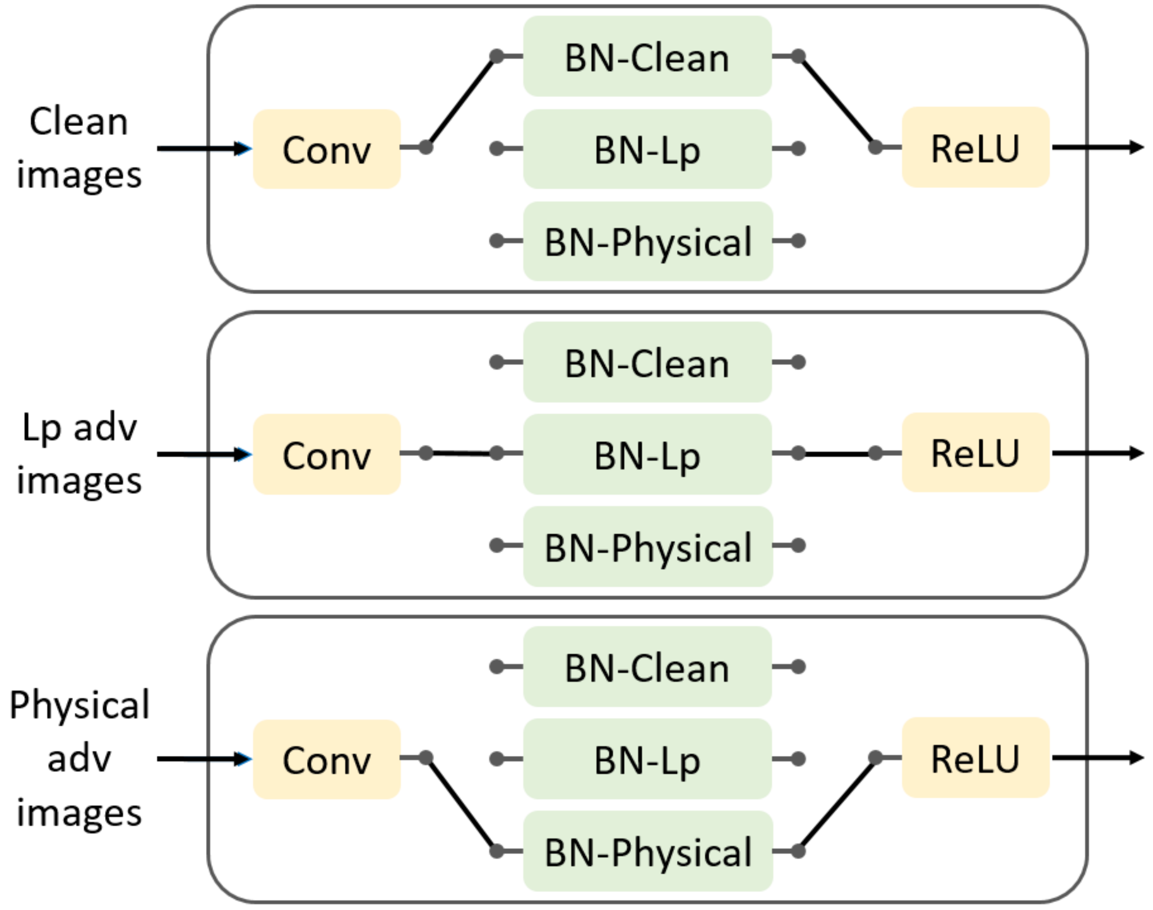
Clean PGD ROA AF SPA

# Extension for Multi-perturbation Robustness

- What about **unforeseen** attack types?
- Example:
  - **Known**: Clean, PGD, ROA
  - **Unforeseen**: AF, SPA
- **Lp-norm attacks**: PGD, SPA
- **Physically realizable attacks**: ROA, AF
- Our assumption: Similar attack types have **similar** data distributions.

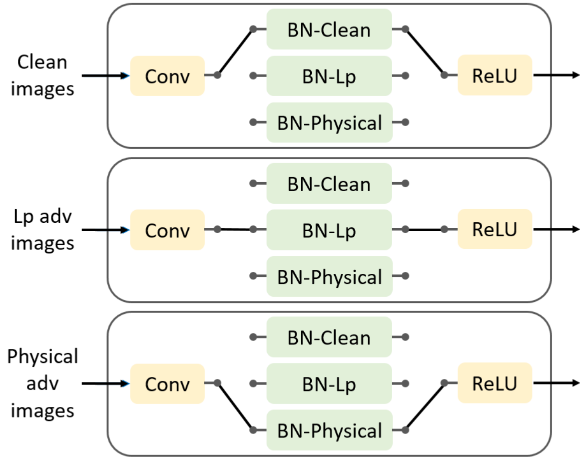
# Our Solution: Multi-BN Structure

- Example:
  - **Known**: Clean, PGD, ROA
  - **Unforeseen**: AF, SPA
- **Lp-norm attacks**: PGD, SPA
- **Physically realizable attacks**: ROA, AF



# Our Solution: Multi-BN Structure

- Training: Clean, $\delta_{PGD}$, $\delta_{ROA}$
- Test: Clean, $\delta_{PGD}$, $\delta_{ROA}$, $\delta_{AF}$, $\delta_{SPA}$




$$\theta = \theta^c + \sum_{i=0}^{N} \theta_i^b$$

$$\underbrace{\theta^* = \arg\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{D}} \left[ \underbrace{L(x, y; \theta^c, \theta_0^b)}_{\text{Clean data}} + \underbrace{\sum_{i=1}^{N} \max_{\delta_i \in \mathbb{S}_i} L(x + \delta_i, y; \theta^c, \theta_i^b)}_{\text{Generate multiple types of adversarial examples}} \right]}_{\text{Train model parameters}}$$

# Our Solution: Multi-BN Structure

| Model | Clean | PGD | ROA | AF | SPA | Mean | Union |
|---|---|---|---|---|---|---|---|
| No Defense | **89.0** | 3.3 | 0.5 | 1.6 | 8.4 | 20.6 | 0.0 |
| AT-PGD | 78.6 | **49.0** | 5.0 | 0.6 | 67.1 | 40.1 | 0.3 |
| AT-ROA | 82.6 | 12.5 | **69.0** | 54.0 | 17.6 | 47.1 | 7.9 |
| AT-AF | 84.6 | 7.1 | 3.9 | **80.5** | 12.2 | 37.7 | 2.1 |
| AT-SPA | 83.5 | 36.9 | 2.6 | 0.7 | **69.5** | 38.6 | 0.2 |
| MultiBN-manual | 83.7 | 46.4 | 65.6 | 57.0 | 60.4 | **62.6** | **40.7** |

# Our Solution: Multi-BN Structure

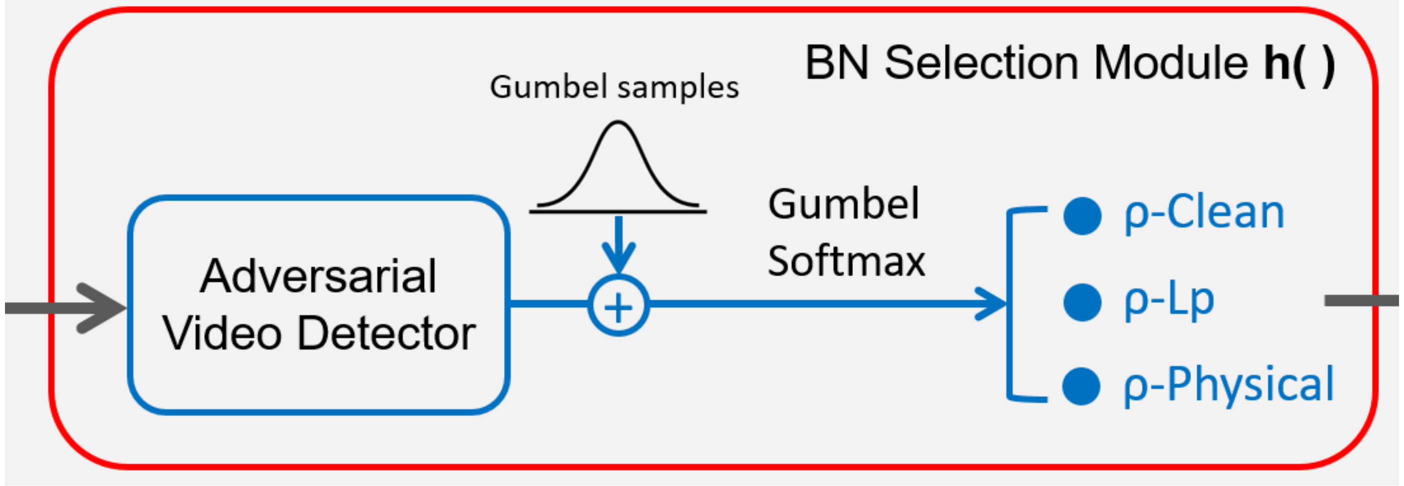
- Performance (%) of each BN branch on the five input types.

| BN Branch | Clean | PGD | ROA | AF | SPA |
|---|---|---|---|---|---|
| BN-Clean | **83.7** | 21.3 | 13.5 | 5.9 | 23.8 |
| BN-Lp | 79.0 | **46.4** | 7.7 | 1.9 | **60.4** |
| BN-Physical | 83.0 | 23.5 | **65.6** | **57.0** | 26.6 |

- Our assumptions are **valid**:
    - Different attack types have **distinct** data distributions.
    - Similar attack types have **similar** data distributions.

# BN Selection Module

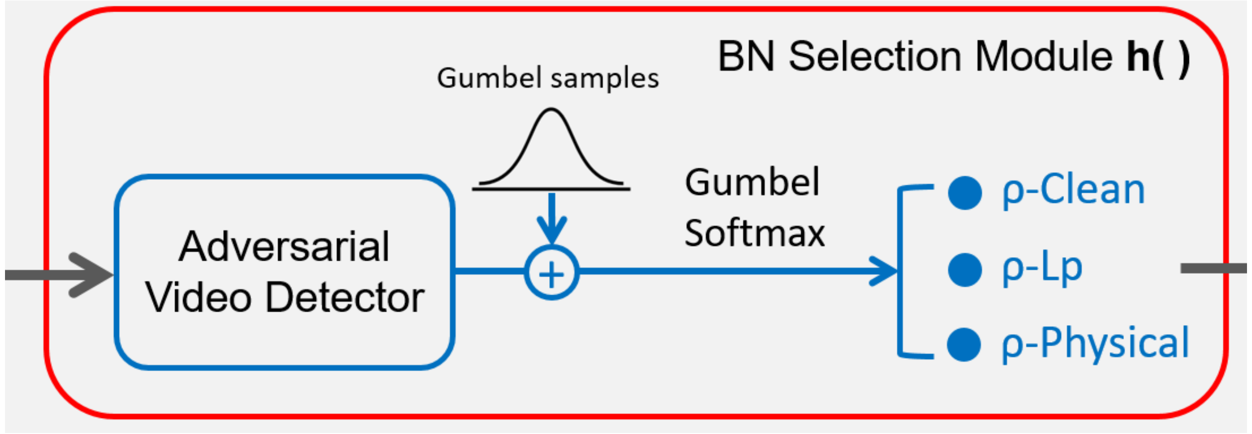
- At inference time, the input data have to pass through the corresponding BN branch **automatically**.
- The **adversarial video detector** is achieved by a video classifier.
- **Gumbel-Softmax** function [Jang et al. ICLR'17] is a **differentiable** approximation of the ***argmax*** operation (vanilla Softmax also works).



# BN Selection Module

- Use Gumbel-Softmax scores as ratio factors to weight each BN branch's output features.

$$\hat{z} = \sum_{k=1}^{K} \rho_k z_k$$




$K: \#\ BN\ branches$

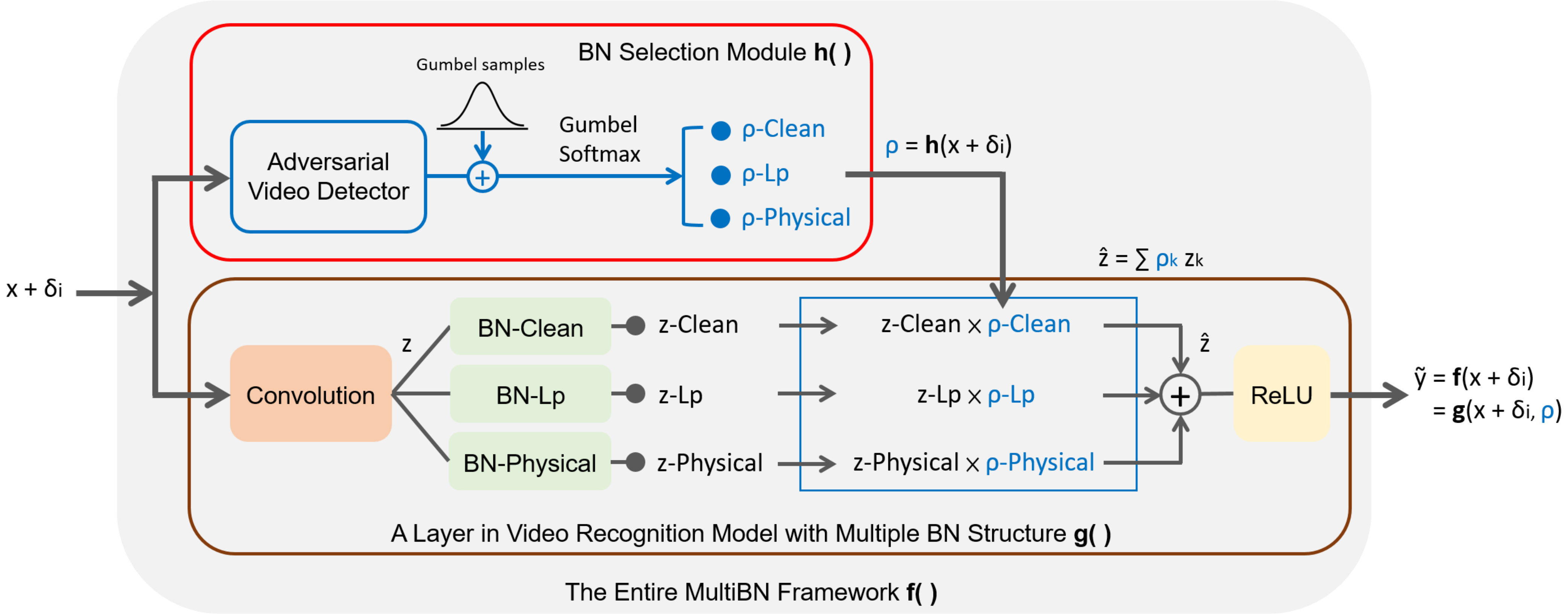
$\rho_1, \dots, \rho_K: ratio\ factors$

$z_1, \dots, z_K: each\ BN\ branch's\ output\ features$

$\hat{z}: weighted\ features$

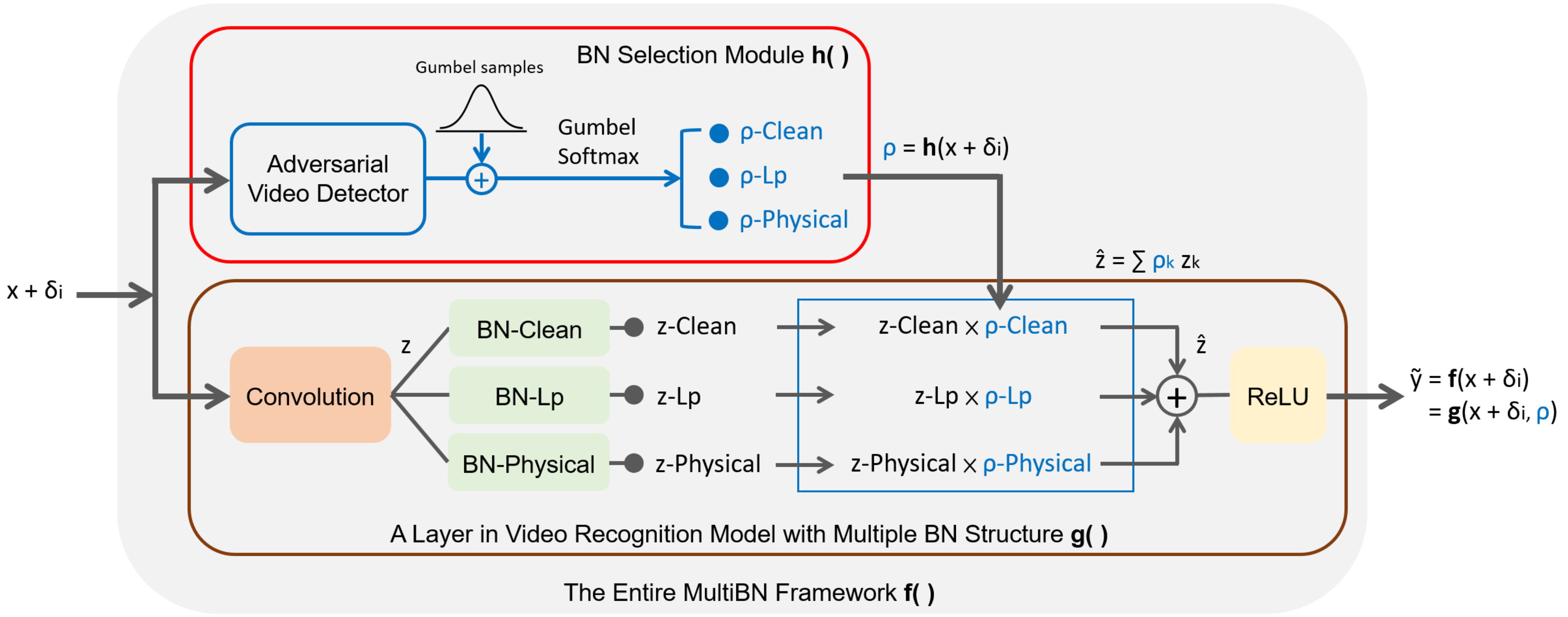
# Entire Framework

- End-to-end pipeline: $\tilde{y} = f(x + \delta_i;\, \theta^c, \theta^b, \theta^{det})$
$$= g(x + \delta_i, h(x + \delta_i;\, \theta^{det});\, \theta^c, \theta^b)$$



# Entire Framework

- End-to-end training: $$\theta^* = \arg\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{D}} \Big[ L(x,y;\theta) + \lambda \cdot L(x, y^{det}; \theta^{det}) + \sum_{i=1}^{N} \big( \max_{\delta_i \in \mathbb{S}_i} L(x+\delta_i, y; \theta) + \lambda \cdot L(x+\delta_i, y^{det}; \theta^{det}) \big) \Big]$$



# Experimental Setup

- Dataset: UCF-101 (action recognition)
- Model: 3D ResNeXt-101
- Attack setting:
  PGD Linf: $\varepsilon=4/255$, T=5
  ROA: patch size=30x30
  AF: width=10
  SPA: #pixels=100, T=5
- White-box attacks
- Untargeted attacks

# Results

Dataset: UCF-101

| Model | Clean | PGD | ROA | AF | SPA | Mean | Union |
|---|---|---|---|---|---|---|---|
| No Defense | **89.0** | 3.3 | 0.5 | 1.6 | 8.4 | 20.6 | 0.0 |
| TRADE [19] (ICML'19) | 82.3 | 29.0 | 5.7 | 3.3 | 42.2 | 32.5 | 1.9 |
| AVG [26] (NeurIPS'19) | 68.9 | 38.1 | 51.4 | 18.5 | 49.6 | 45.3 | 17.3 |
| MAX [26] (NeurIPS'19) | 72.8 | 32.5 | 31.0 | 5.8 | 49.4 | 38.3 | 5.5 |
| MSD [27] (ICML'20) | 70.2 | 43.2 | 1.7 | 1.6 | **56.0** | 34.6 | 0.7 |
| MultiBN (ours) | 74.2 | **44.6** | **58.6** | **44.3** | 53.7 | **55.1** | **34.8** |

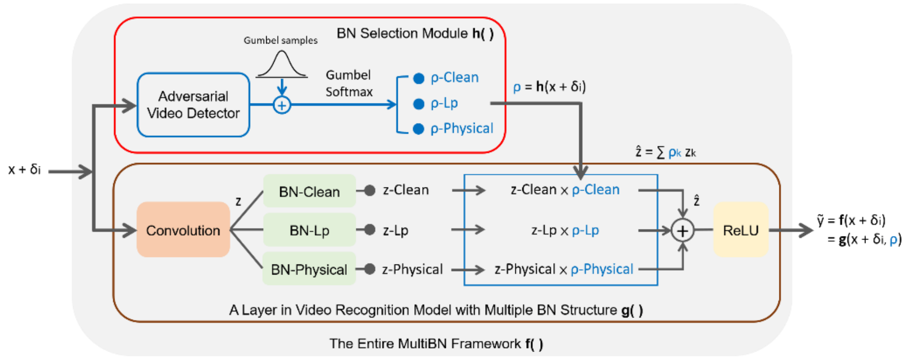
Dataset: HMDB-51

| Model | Clean | PGD | ROA | AF | SPA | Mean | Union |
|---|---|---|---|---|---|---|---|
| No Defense | **65.1** | 0.0 | 0.0 | 0.0 | 0.3 | 13.1 | 0.0 |
| TRADE [19] (ICML'19) | 54.8 | 6.8 | 0.3 | 0.0 | 20.5 | 16.5 | 0.0 |
| AVG [26] (NeurIPS'19) | 39.0 | 14.3 | 17.1 | 2.8 | 26.2 | 19.9 | 1.4 |
| MAX [26] (NeurIPS'19) | 48.6 | 13.9 | 16.0 | 0.1 | 30.3 | 21.8 | 0.0 |
| MSD [27] (ICML'20) | 41.4 | 18.2 | 0.1 | 0.0 | **31.2** | 18.2 | 0.0 |
| MultiBN (ours) | 51.1 | **22.0** | **23.7** | **7.8** | 29.9 | **26.9** | **5.0** |

# Results: Robustness Against Adaptive Attacks

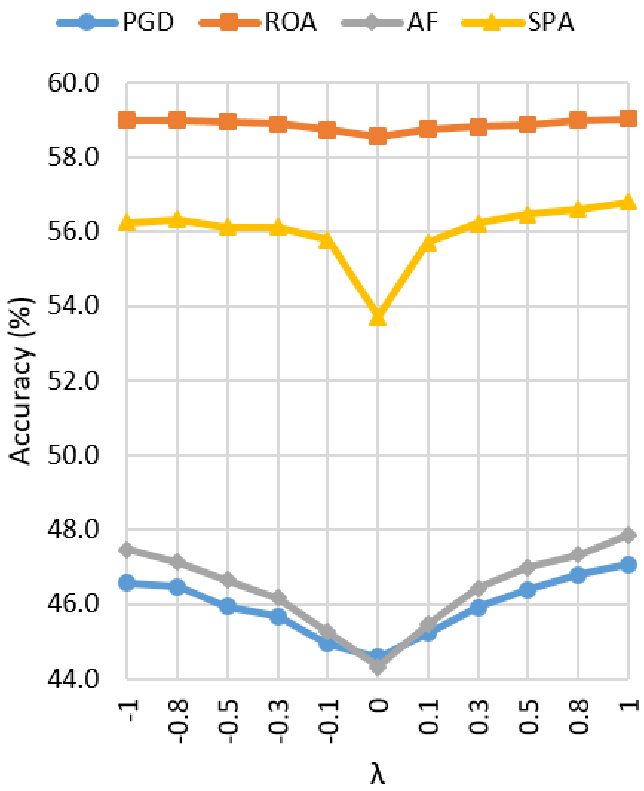
- Construct an adaptive attack, which jointly attacks the **target model part** and the **BN selection module part**.
- The intuition is to generate adversarial examples which can also fool the BN selection module to let it select the incorrect BN branch, and thus become easier to fool the target model.

$$\delta = \arg\max_{\delta \in \mathbb{S}} \left[ L(x+\delta, y; \theta) + \lambda \cdot L(x+\delta, y^{det}; \theta^{det}) \right]$$



# Results: Robustness Against Adaptive Attacks

- The canonical attack has the greatest attacking strength.
- The proposed MultiBN is robust against adaptive attacks.



# Results: Different Attack Budget

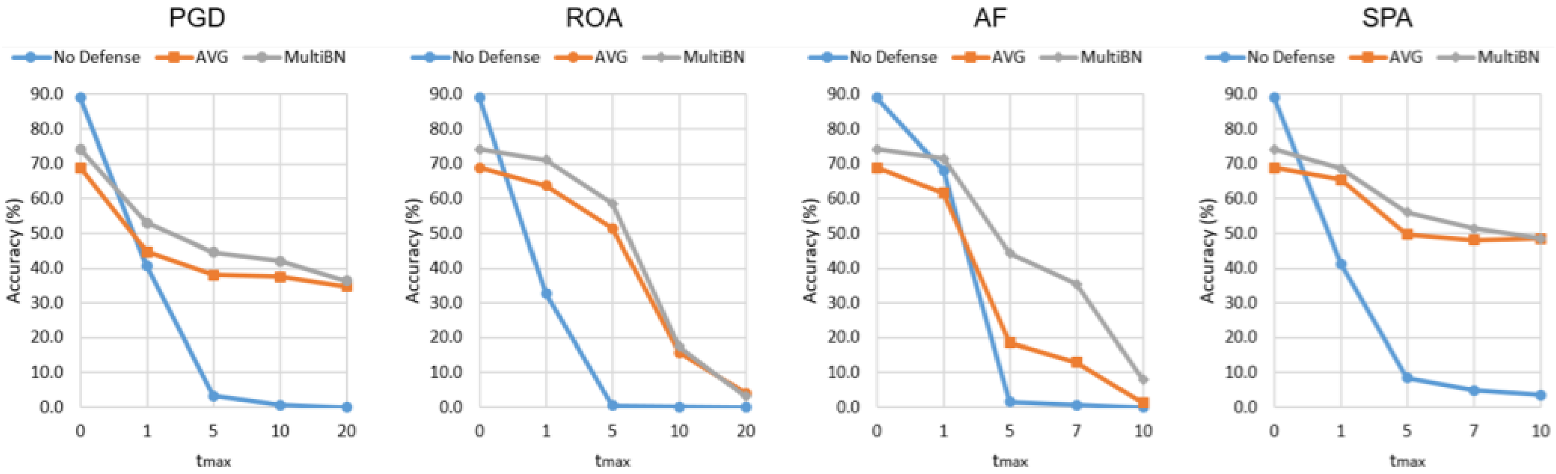



Fig. 3: Results (%) under the four attack types with varied numbers of attack iterations $t_{max}$.

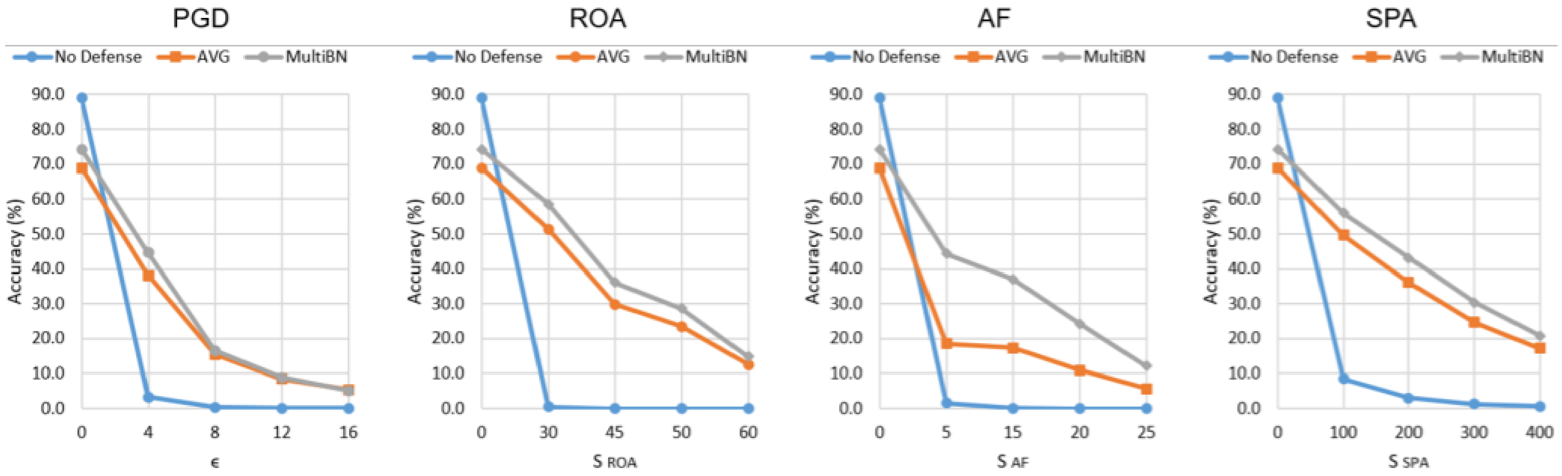



Fig. 4: Results (%) under the four attack types with varied perturbation bounds.

# Results: Robustness Against Black-box Attacks

- Generate adversarial videos on a **surrogate** model: 3D Wide ResNet-50
- Test on the **target** model: 3D ResNeXt-101

| Model | Clean | PGD | ROA | AF | SPA | Union |
|---|---|---|---|---|---|---|
| TRADE [23] (ICML'19) | **82.3** | **81.0** | 60.8 | <u>65.0</u> | **78.0** | 49.3 |
| AVG [30] (NeurIPS'19) | 68.9 | 68.4 | 68.0 | 62.0 | 68.4 | 56.2 |
| MAX [30] (NeurIPS'19) | 72.8 | 72.4 | <u>71.4</u> | 63.5 | <u>71.9</u> | <u>57.9</u> |
| MSD [31] (ICML'20) | 70.2 | 69.8 | 40.1 | 52.2 | 69.1 | 31.3 |
| MultiBN (ours) | <u>74.2</u> | <u>73.6</u> | **74.0** | **72.4** | 71.5 | **63.5** |

# Results on Images

- Dataset: CIFAR-10
- Model: ResNet-18

| Model | Clean | PGD | ROA | AF | SPA | Mean | Union |
|---|---|---|---|---|---|---|---|
| No Defense | **94.3** | 0.0 | 4.7 | 0.1 | 16.3 | 23.1 | 0.0 |
| TRADE [23] (ICML'19) | 71.4 | 14.7 | 34.7 | 30.4 | 52.8 | 40.8 | 10.1 |
| AVG [30] (NeurIPS'19) | 86.4 | 47.2 | 53.6 | 60.5 | <u>67.8</u> | 63.1 | 28.1 |
| MAX [30] (NeurIPS'19) | 87.7 | 46.3 | 60.0 | 54.6 | **73.6** | <u>64.4</u> | <u>33.7</u> |
| MSD [31] (ICML'20) | 93.0 | **52.7** | 6.7 | 7.1 | 59.6 | 43.8 | 2.2 |
| MultiBN (ours) | <u>94.2</u> | <u>49.7</u> | **74.9** | **66.7** | 60.9 | **69.3** | **36.9** |

# Thanks for your attention