



Yinpeng Dong^{1,3}, Shouwei Ruan², Hang Su¹, Caixin Kang², Xingxing Wei², Jun Zhu^{1,3}

¹ Dept. of Comp. Sci. and Tech., Institute for AI, Tsinghua University

² Institute of AI, Beihang University ³ RealAI

dongyinpeng@mail.tsinghua.edu.cn

OOD Generalization is Hard

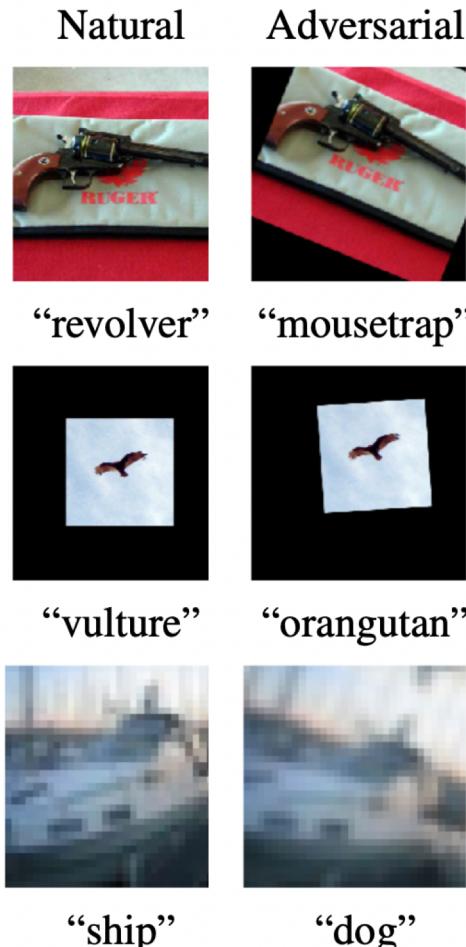


Image Translation and
Rotation (Engstrom et
al., 2019)

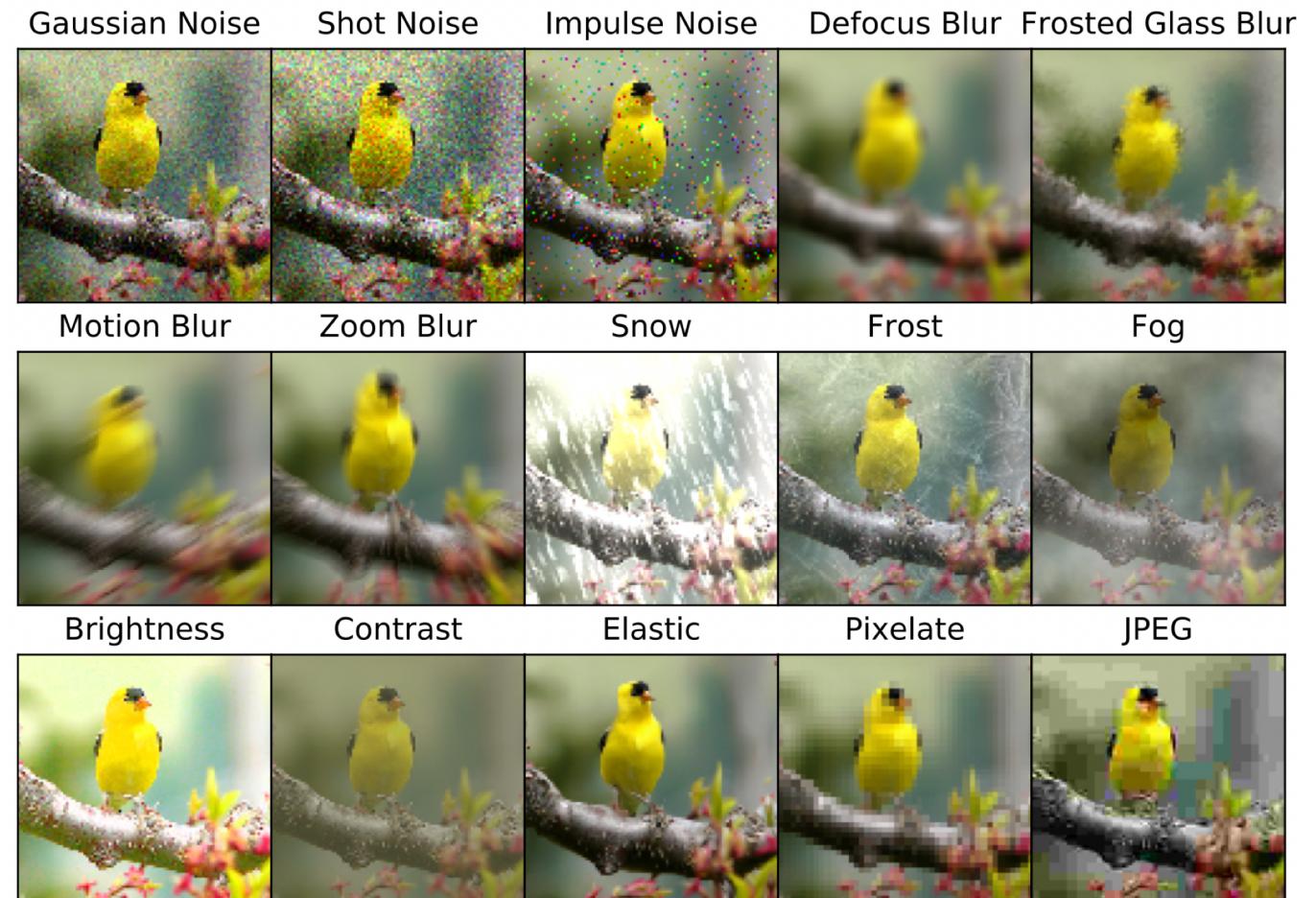


Image Corruptions (Hendrycks
et al., 2019)

Viewpoint Changes



Chair: 78.66%



Keyboard: 47.80%



Street sign: 99.55%



Traffic light: 97.94%



Board: 63.50%



Mouse: 37.44%



Cinema: 58.45%



Canoe: 41.01%

Goal: find **adversarial viewpoints** that lead to wrong predictions of visual recognition models in the **physical world**.

Real-world Application



Autonomous driving fails to recognize trucks/cars in rare viewpoints to cause traffic accidents.

Challenges

- How to model real-world 3D objects with high-fidelity? — **NeRF**
 - Simple pipeline: 1) training a NeRF; 2) optimize viewpoint parameters based on NeRF; 3) verify the vulnerability from the adversarial viewpoint
- How to mitigate the **reality gap** between a real object and its neural representation?

NeRF
rendering



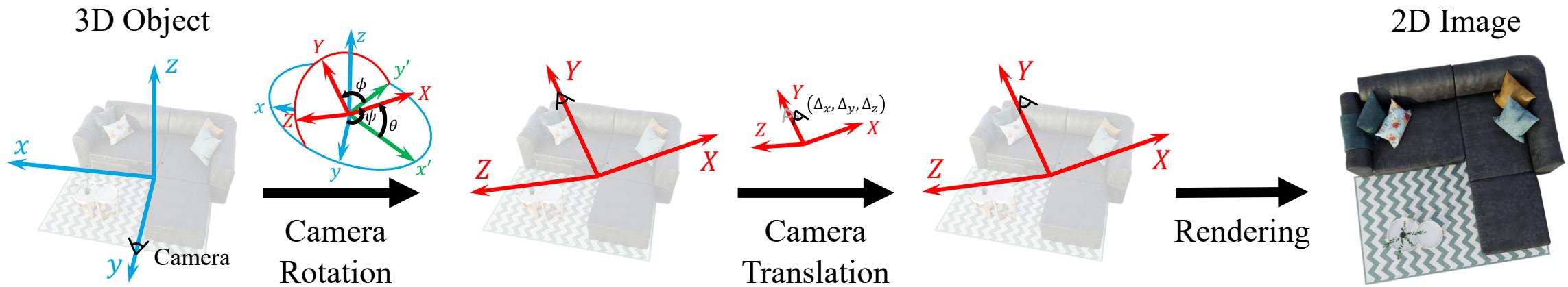
Real
object



- How to **control the real camera pose** to precisely match the adversarial viewpoint?

ViewFool: Problem Formulation

- Let $\mathbf{v} := [\psi, \theta, \phi, \Delta_x, \Delta_y, \Delta_z]$ denote the transformation parameters of the camera.
- Let $I := R(\mathbf{v})$ denote the rendered image.



ViewFool: Optimization Problem

- Learning a **distribution** of adversarial viewpoints:

$$\max_{p(\mathbf{v})} \left\{ \mathbb{E}_{p(\mathbf{v})} [L(f(R(\mathbf{v})), y)] + \lambda \cdot H(p(\mathbf{v})) \right\}$$

- Since $\max_{p(\mathbf{v})} \mathbb{E}_{p(\mathbf{v})} [L(f(R(\mathbf{v})), y)] \leq \max_{\mathbf{v}} L(f(R(\mathbf{v})), y)$, we add an **entropic regularizer** into the objective as

$$H(p(\mathbf{v})) = -\mathbb{E}_{p(\mathbf{v})} [\log p(\mathbf{v})]$$

ViewFool: Optimization Algorithm

- A transformation of Gaussian distribution

$$\mathbf{v} = \mathbf{a} \cdot \tanh(\mathbf{u}) + \mathbf{b}; \mathbf{u} \sim N(\boldsymbol{\mu}, \sigma^2 \mathbf{I}), \mathbf{a} = \frac{\mathbf{v}_{max} - \mathbf{v}_{min}}{2}, b = \frac{\mathbf{v}_{max} + \mathbf{v}_{min}}{2}$$

- Our problem becomes

$$\max_{\boldsymbol{\mu}, \sigma} \left\{ \mathbb{E}_{N(\mathbf{u}; \boldsymbol{\mu}, \sigma^2 \mathbf{I})} [L(f(R(\mathbf{a} \cdot \tanh(\mathbf{u}) + \mathbf{b})), y) - \lambda \cdot \log p(\mathbf{a} \cdot \tanh(\mathbf{u}) + \mathbf{b})] \right\}$$

- Reparameterization trick: $\mathbf{u} = \boldsymbol{\mu} + \sigma \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \mathbf{I})$.

- Gradient calculation: adopt the search gradients

$$\nabla_{\boldsymbol{\mu}} = \mathbb{E}_{\mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})} [\mathcal{L}(f(\mathcal{R}(\mathbf{a} \cdot \tanh(\boldsymbol{\mu} + \sigma \boldsymbol{\epsilon}) + \mathbf{b})), y) \cdot \sigma \boldsymbol{\epsilon} - \lambda \cdot 2 \tanh(\boldsymbol{\mu} + \sigma \boldsymbol{\epsilon})], \quad (6)$$

$$\nabla_{\sigma} = \mathbb{E}_{\mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})} \left[\mathcal{L}(f(\mathcal{R}(\mathbf{a} \cdot \tanh(\boldsymbol{\mu} + \sigma \boldsymbol{\epsilon}) + \mathbf{b})), y) \cdot \frac{\sigma(\boldsymbol{\epsilon}^2 - 1)}{2} - \lambda \cdot \frac{1 - 2 \tanh(\boldsymbol{\mu} + \sigma \boldsymbol{\epsilon}) \cdot \sigma \boldsymbol{\epsilon}}{\sigma} \right].$$

Visualization of Adversarial Viewpoints

Real Image from
Natural Viewpoint



Granny Smith: 83.71% Studio Couch : 97.56%

Warplane: 94.42%

Notebook: 72.50%

Traffic Light: 99.91%

Forklift: 97.27%

Street Sign: 95.00%

Rendered Image from
Adversarial Viewpoint



Tennis Ball: 97.81%

Wallet: 90.24%

Hatchet: 81.23%

Scale: 93.82%

Mortarboard: 91.97%

Rocking Chair: 58.98%

Spatula: 40.98%

Real Image from
Adversarial Viewpoint



Tennis Ball: 73.09%

Wallet: 93.13%

Hatchet: 31.01%

Scale: 45.23%

Mortarboard: 86.48%

Folding Chair: 36.51%

Street Sign: 11.74%

Cross-model Transferability

	ViewFool	VGG-16	Inc-v3	IncRes-v2	DN-121	EN-B0	MN-v2	DeiT-B	Swin-B	Mixer-B
ResNet-50	$\lambda = 0$	85.00%	75.94%	80.59%	73.97%	76.73%	76.77%	65.22%	55.81%	87.07%
	$\lambda = 0.01$	86.52%	82.00%	82.00%	79.07%	82.62%	79.11%	69.35%	59.62%	90.37%
ViT-B/16	$\lambda = 0$	82.35%	76.18%	76.62%	74.62%	77.06%	72.14%	69.34%	60.50%	87.80%
	$\lambda = 0.01$	82.83%	78.73%	79.07%	77.45%	74.92%	73.97%	69.45%	59.01%	85.72%

High transferability between different models!

Real-world Experiments



Real Image from
Natural Viewpoint



Chair: 78.66%

Rendered Image from
Adversarial Viewpoint



Board: 79.53%

Real Image from Adversarial Viewpoint



Board: 63.50%



Board: 57.94%



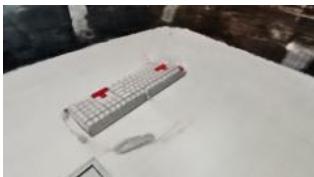
Board: 61.17%



Board: 53.80%



Keyboard: 47.80%



Matchstick: 25.07%



Mouse: 31.27%



Mouse: 25.88%



Mouse: 35.32%



Mouse: 37.44%



Street sign: 99.55%



Mailbox: 18.84%



Cinema: 58.45%



Gas pump : 36.63%



Unicycle: 15.79%



Flagpole: 31.98%



Traffic light: 97.94%



Syringe: 10.03%



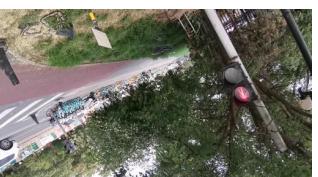
Canoe: 41.01%



Stretcher: 50.01%

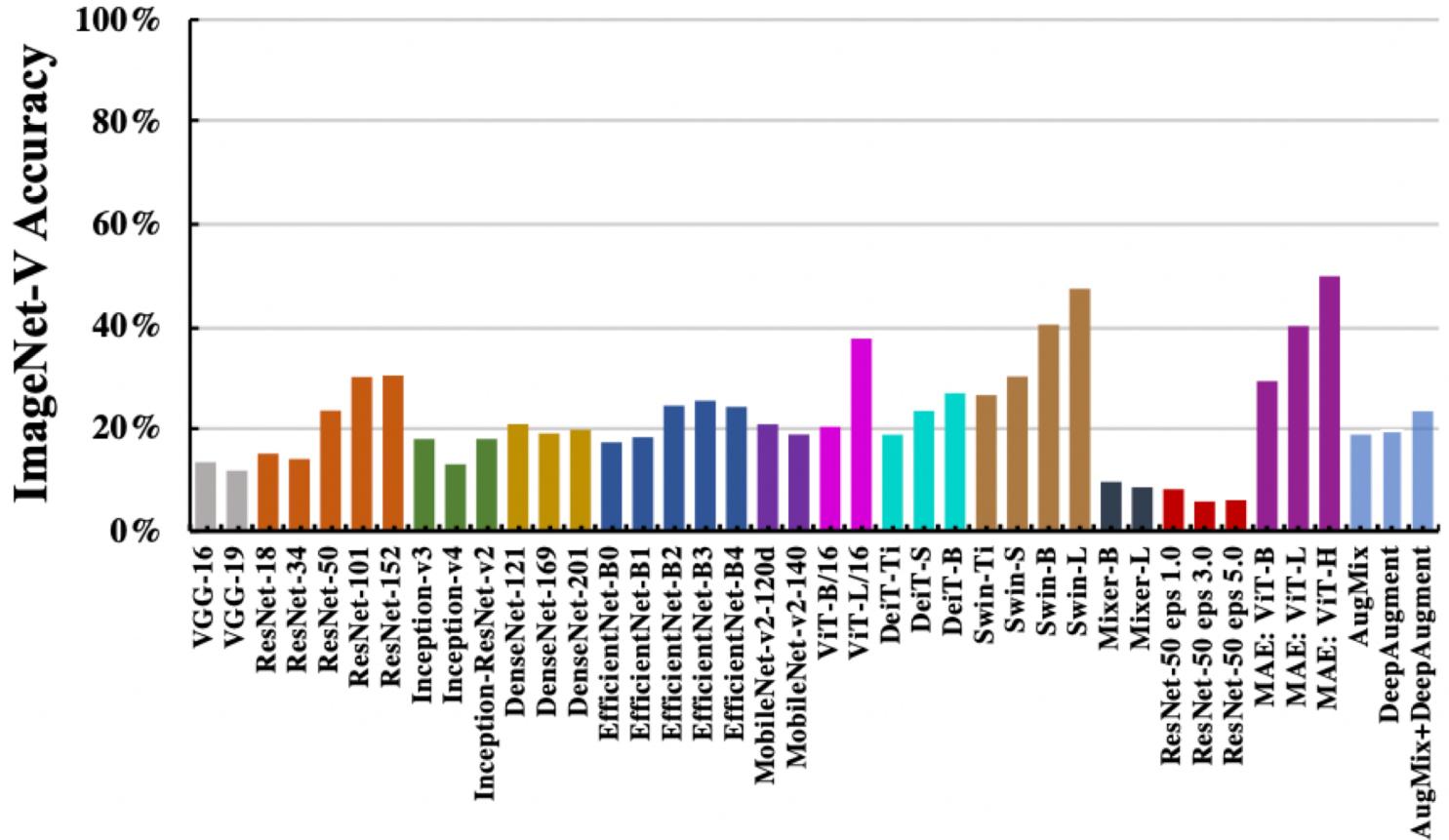


Missile: 8.09%



Solar dish: 31.23%

ImageNet-V Benchmark



- **Transformer-based** models have better viewpoint robustness;
- A **larger model** within the same architecture family tends to perform better;
- **Adversarial training and existing data augmentation** techniques do *not* obtain good results.



Thanks for listening

Code is available at:
https://github.com/Heathcliff-saku/ViewFool_

