On the Robustness and Interpretability of Deep Learning Models

WU, Weibin

Ph.D. Oral Defense

Supervisor: Prof. Michael R. Lyu & Prof. Irwin King

2021/08/31

香港中文大學
The Chinese University of Hong Kong

Wide Deployment of Deep Learning

- Safety- and security-critical domain

Self-driving

Medical Diagnosis

Robustness and Interpretability Are Important

- AI failure

Robustness Issue

JULY 27, 2018

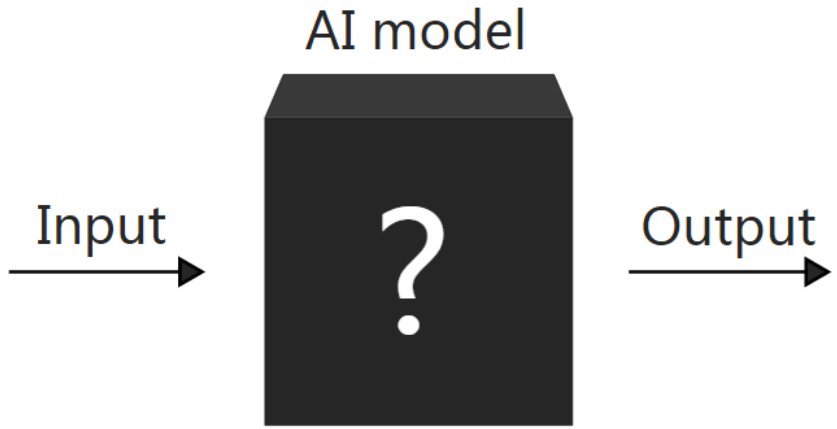
IBM's Watson recommended 'unsafe and incorrect' treatments for cancer patients, investigation reveals

Daily Briefing

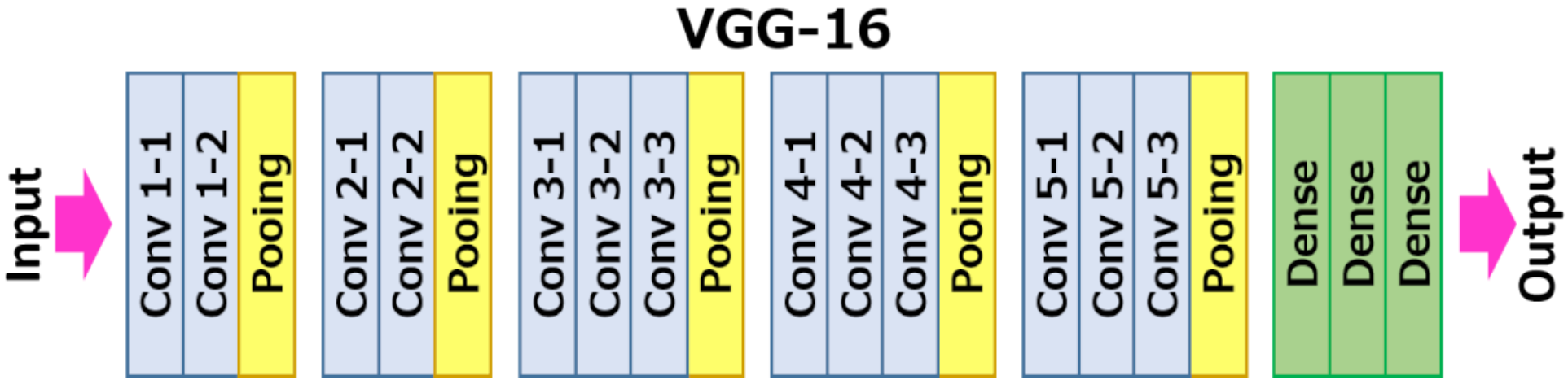
Interpretability Issue

Challenge

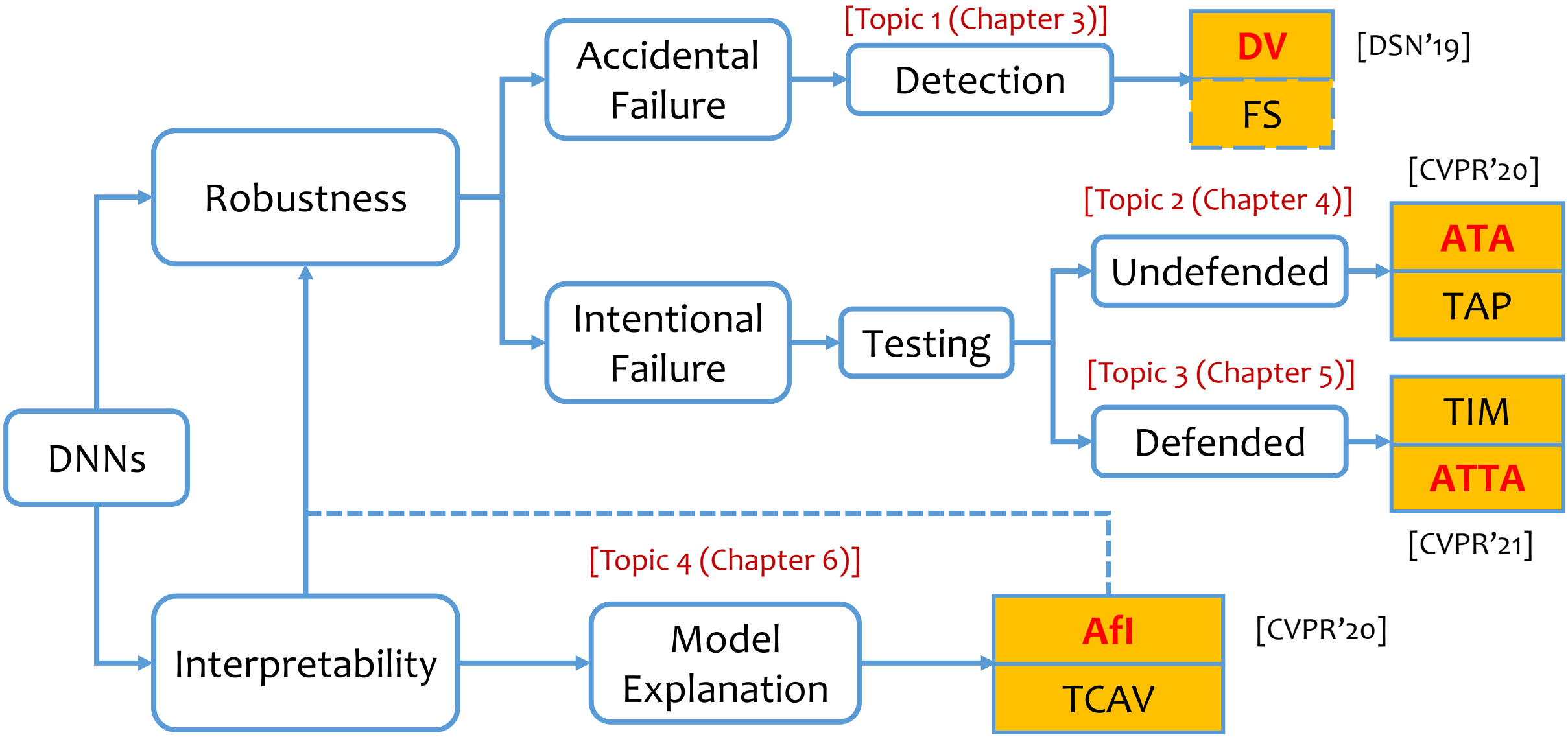
- Black-box nature
  - End-to-end training



- Complexity
  - **VGG16**: 138 million parameters
  - **AmoebaNet-B**: 557 million parameters
  - **BERT-large**: 340 million parameters



Contribution

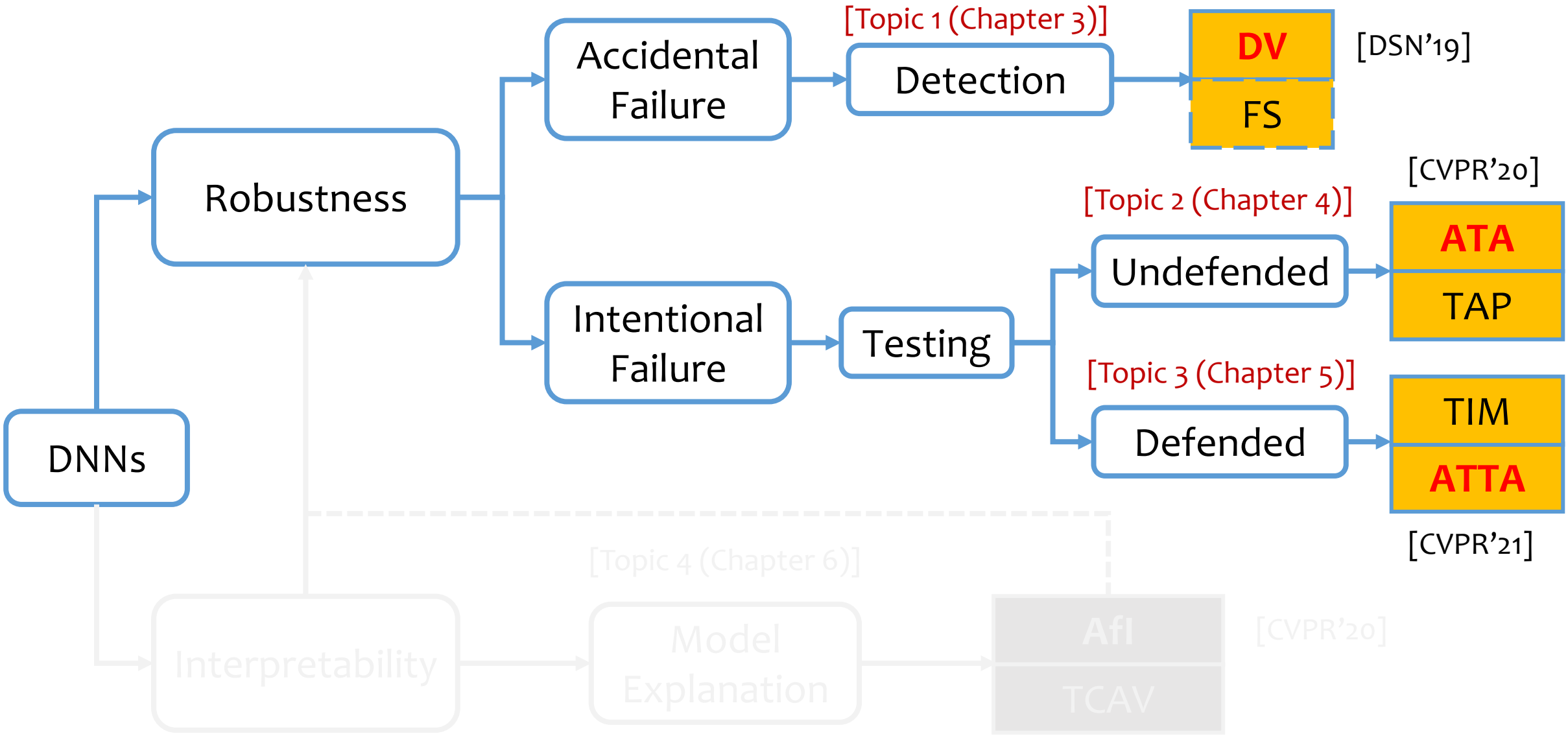


Robustness of DNNs

- Robustness
  - "The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions" (IEEE Std 610.12-1990)
  - We focus on the robustness of DNNs against invalid inputs

- Invalid input
  - Invalid inputs for a deep learning model are the samples that do not come from the training data distribution of the model
    - Real-world corner case → accidental failure
    - Adversarial sample → intentional failure

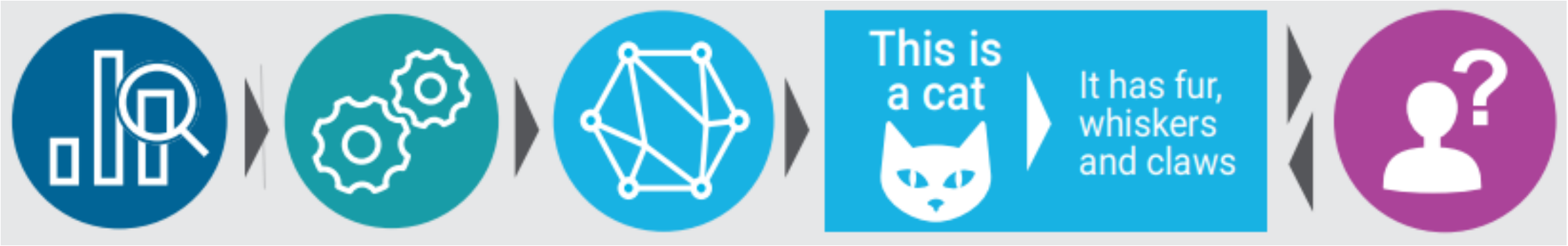
Robustness of DNNs



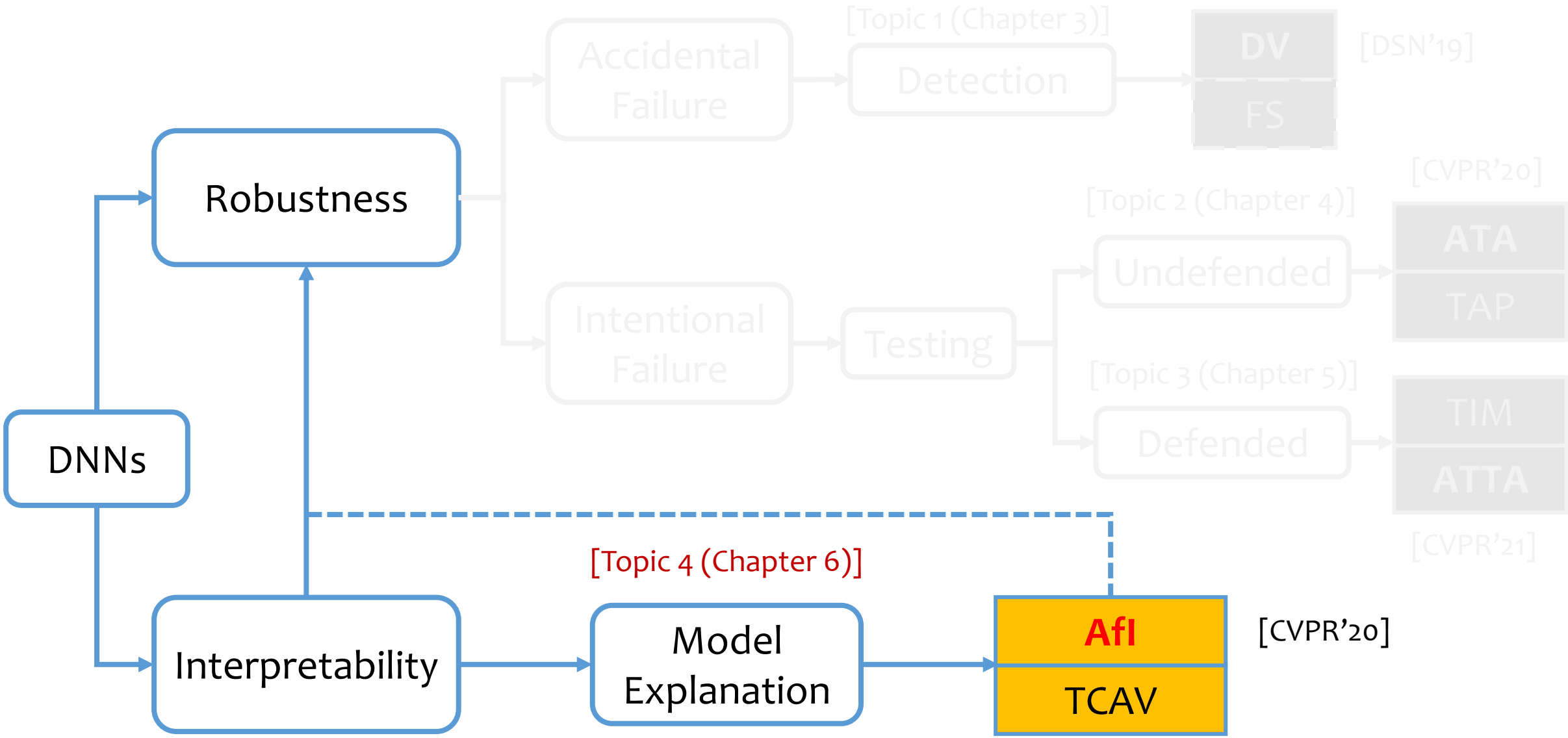
Interpretability of DNNs

- Interpretability
  - "Interpretability is the degree to which a human can understand the cause of a decision" [Molnar, 2020]

- Model explanation
  - Reveal the ground of a model's decision



Interpretability of DNNs



Outline

- Topic 1: Detecting Real-world Corner Cases for DNNs
- Topic 2: Synthesizing Adversarial Samples against Undefended DNNs
- Topic 3: Synthesizing Adversarial Samples against Defended DNNs
- Topic 4: Global Explanations of DNNs
- Conclusion and Future Work

Outline

- Topic 1: Detecting Real-world Corner Cases for DNNs
- Topic 2: Synthesizing Adversarial Samples against Undefended DNNs
- Topic 3: Synthesizing Adversarial Samples against Defended DNNs
- Topic 4: Global Explanations of DNNs
- Conclusion and Future Work

Motivation

- Real-world corner case
  - Naturally occurred, but often unusual samples that are overlooked during the design of the system
  - Accidental failure

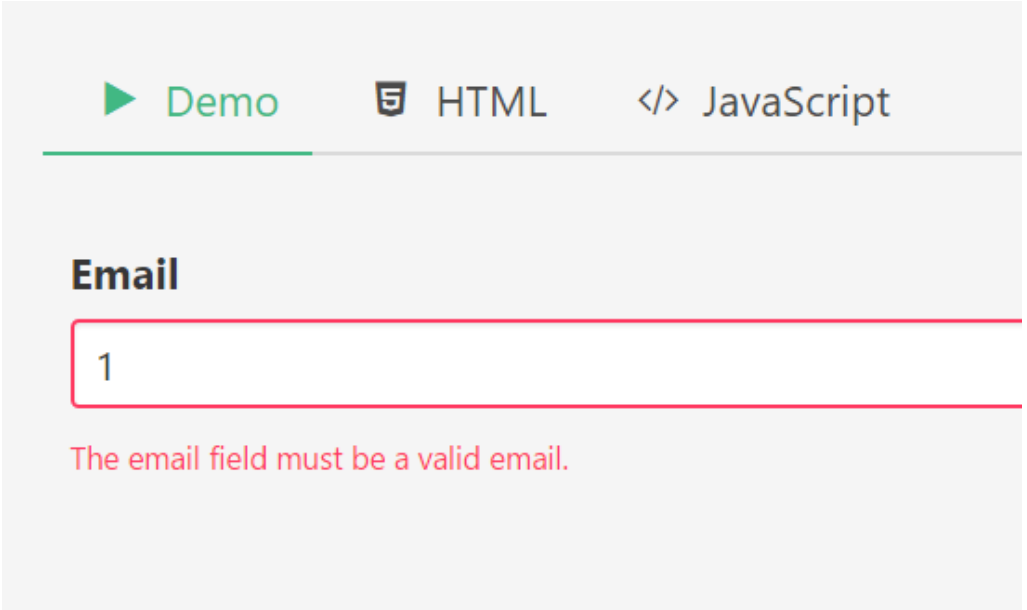
[Zhang et al., 2018]

Motivation

- Existing effort
  - Testing: synthesize real-world corner cases
  - Debugging: fix the discovered failures
  - Drawback: limited data during development vs. innumerable working conditions during deployment
- Detection
  - Ensure DNNs' correct functionality during deployment
  - Enable fail-safe action
- Research question
  - How can we detect real-world corner cases?

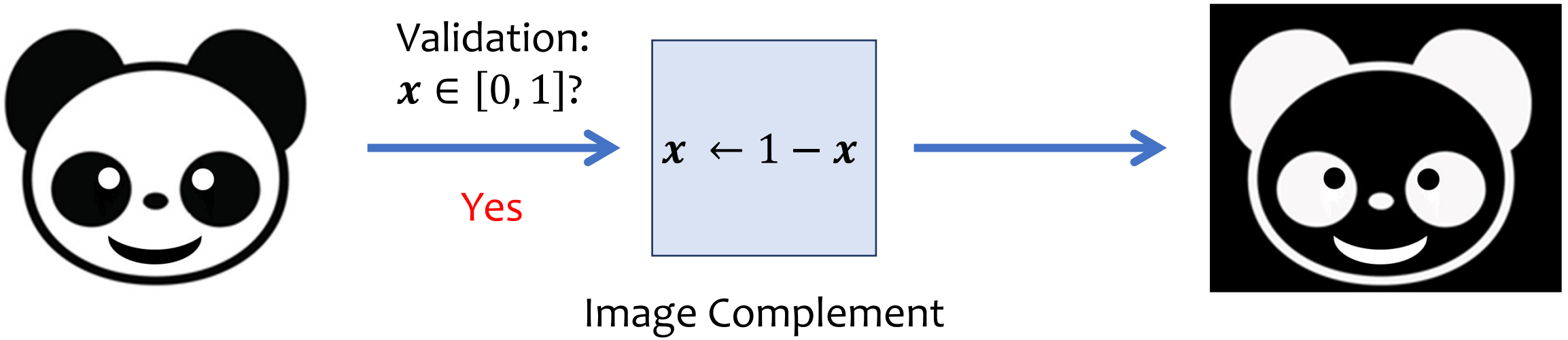
Method

- Motivation
  - DNNs vs. traditional software
- Real-world corner case
  - Invalid inputs that exceed the capacity of DNNs
- Input validation for traditional software
  - Ensure only valid data can enter the system



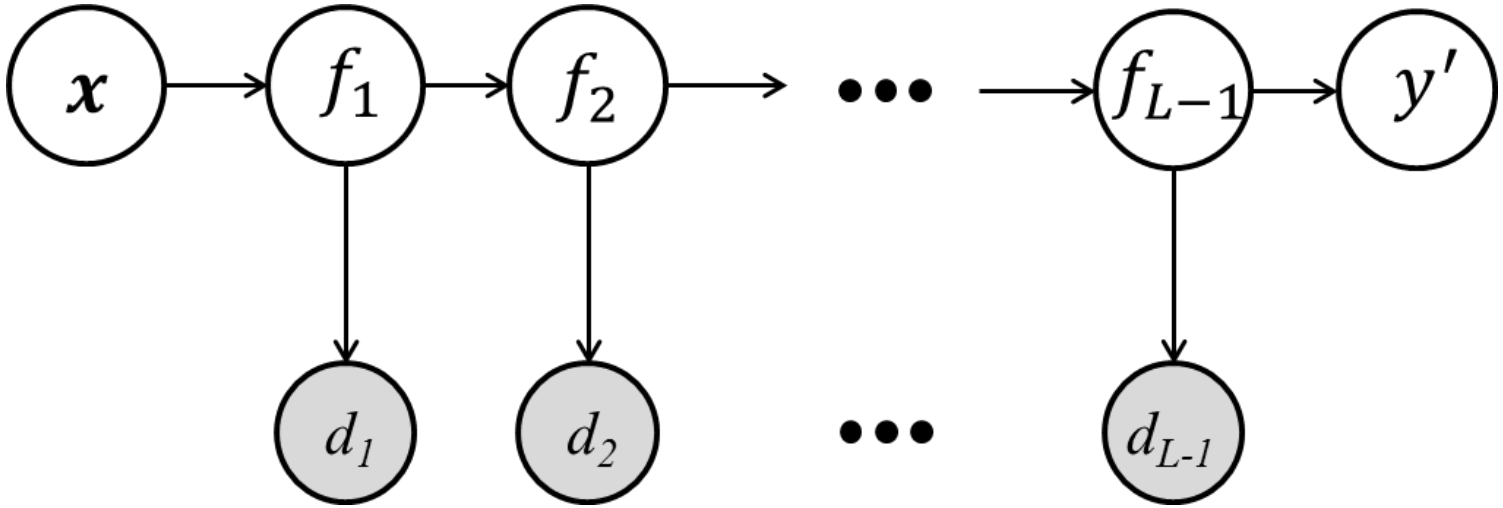
Challenge

- DNNs vs. traditional software
  - Unlike traditional programs, DNNs' function is learned automatically from the training data, instead of being programmed by developers
  - How to model the specification of DNNs and derive the validation rules?



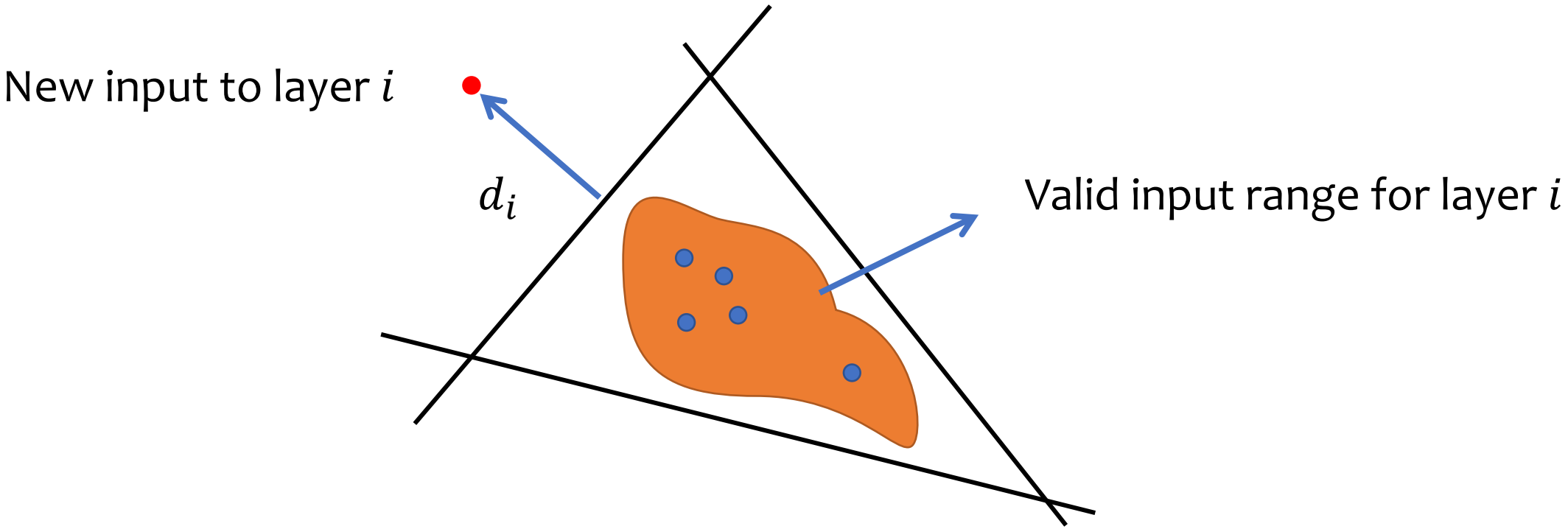
Method

- Deep Validation (DV)
  - Each layer performs relatively simple functions
  - Validate the inputs of each layer
    - f_i : output of the i-th layer
    - d_i: estimate the discrepancy of the input of layer i to its valid range



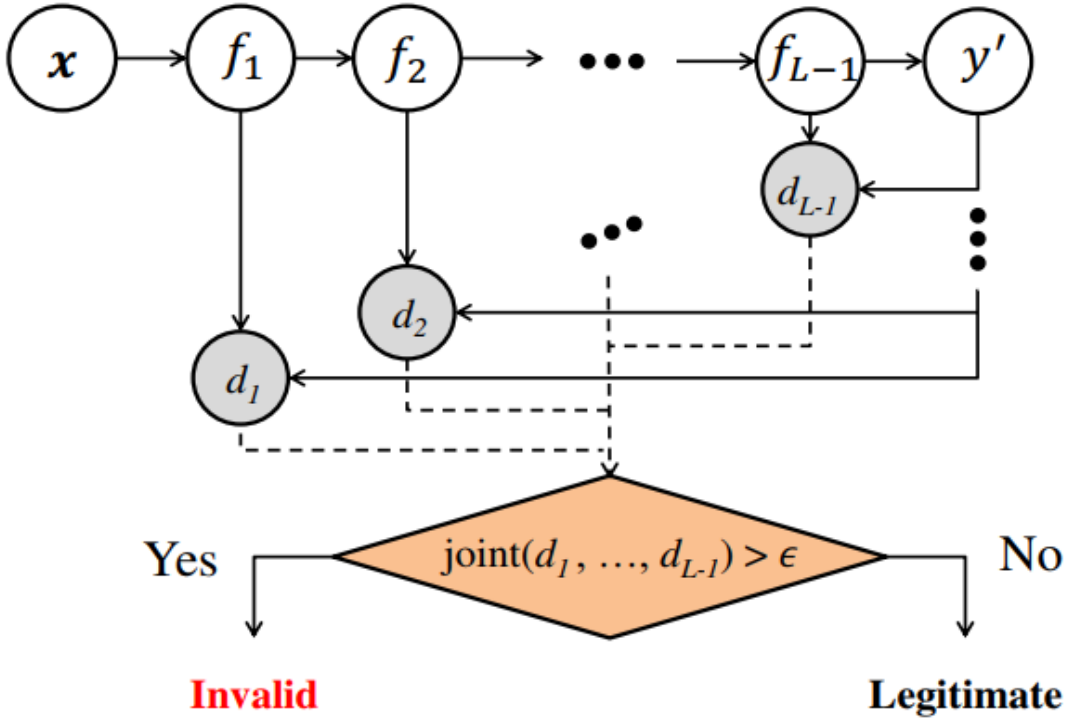
Deep Validation

- How to compute d_i?
  - Resort to the training data: one-class SVM
    - Only need valid samples
    - d_i: signed distance to the learned separating hyperplane



Deep Validation

- Framework
  - d_i: signed distance to the learned separating hyperplane in layer i → single validator in layer i
  - joint$(d_1, \ldots, d_{L-1}) = \sum d_i$ → joint validator



Experiments

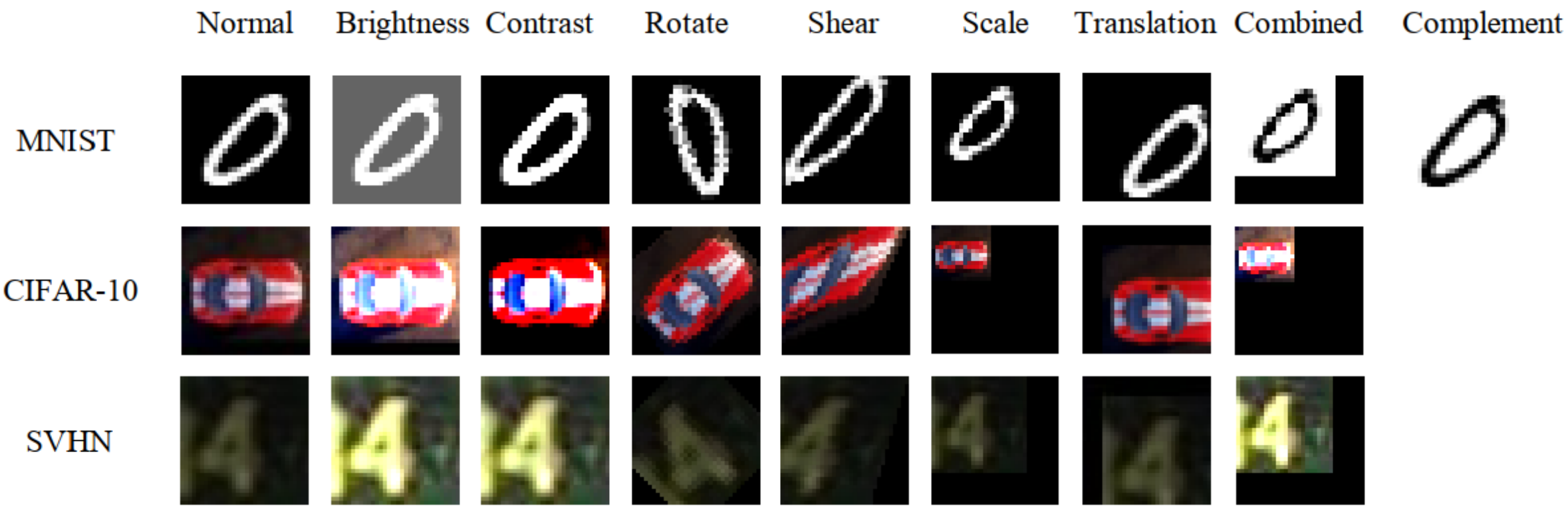
- Dataset
  - MNIST: seven-layer CNN
  - CIFAR-10: DenseNet
  - SVHN: seven-layer CNN

- Baseline
  - Adversarial sample detection method
    - Feature Squeezing
    - Kernel Density Estimation

- Metric: ROC-AUC score (↑)
  - Reflect both the true positive rate (TPR) and the false positive rate (FPR)
  - Higher ROC-AUC score → better detection performance

Experiments

- Synthesize real-world corner cases
  - Metamorphic testing technique
  - Over 65.8% misclassification



Experiments

- Do adversarial sample detection methods really capture the valid input range of DNNs? – **No**
  - SCCs: only view successful corner cases that can cause misclassification as true positives

Dataset	Method	Overall ROC-AUC Score (SCCs)
MNIST	Deep Validation	**0.9937**
	Feature Squeezing	0.9784
	Kernel Density Estimation	0.1436
CIFAR-10	Deep Validation	**0.9805**
	Feature Squeezing	0.8796
	Kernel Density Estimation	0.1254
SVHN	Deep Validation	**0.9506**
	Feature Squeezing	0.6870
	Kernel Density Estimation	0.2543

Experiments

- Single validator vs. joint validator (MNIST as an example)
  - A joint validator often provides additional gains

Configuration		Transformation Method Used to Synthesize Corner Cases						Overall ROC-AUC Score (SCCs)
Validator	Layer No.	Rotation	Shear	Scale	Translation	Complement	Combined	
Single Validator	1	0.8760	0.9987	0.8827	0.8952	**1.0000**	**1.0000**	0.9440
	2	0.9200	0.9719	0.8048	0.8893	**1.0000**	0.9996	0.9324
	3	0.9741	0.9797	0.9591	0.9728	0.9850	0.9197	0.9618
	4	0.9740	0.9823	0.9224	0.9657	0.9876	0.9670	0.9657
	5	0.9732	0.9788	0.9053	0.9602	0.9861	0.9630	0.9601
	6	0.9659	0.9889	0.9237	0.9620	0.9871	0.9786	0.9676
Best Transformation-specific Single Validator		3	1	3	3	1, 2	1	6
Joint Validator		**0.9891**	**0.9991**	**0.9881**	**0.9844**	**1.0000**	**1.0000**	**0.9937**

Experiments

- Can Deep Validation also spot adversarial samples as invalid inputs?
  - **Yes, with great promise**
    - SAEs: only view successful adversarial examples as true positives
    - AEs: view all adversarial examples as true positives

Attack Method		FGSM	BIM	CW_∞		CW_2		CW_0		JSMA		Overall ROC-AUC Score
Target Label		-	-	Next	LL	Next	LL	Next	LL	Next	LL	
Success Rate		0.4300	0.9100	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.6650	0.5150	
SAEs	Deep Validation	1.0000	1.0000	0.9992	0.9965	0.9347	0.9758	0.9329	0.9651	0.9851	0.9944	0.9755
	Feature Squeezing	0.9970	0.9972	1.0000	1.0000	0.9993	0.9996	0.9920	0.9920	0.9973	0.9972	**0.9971**
AEs	Deep Validation	1.0000	1.0000	0.9992	0.9965	0.9347	0.9758	0.9329	0.9651	0.9282	0.8399	**0.9572**
	Feature Squeezing	0.9441	0.9691	1.0000	1.0000	0.9993	0.9996	0.9920	0.9920	0.8169	0.6870	0.9400

Summary

1. We introduce Deep Validation as the first framework to automatically validate internal inputs and detect real-world corner cases for DNNs

2. We conduct extensive experiments to confirm the superior performance of Deep Validation to state-of-the-art baselines

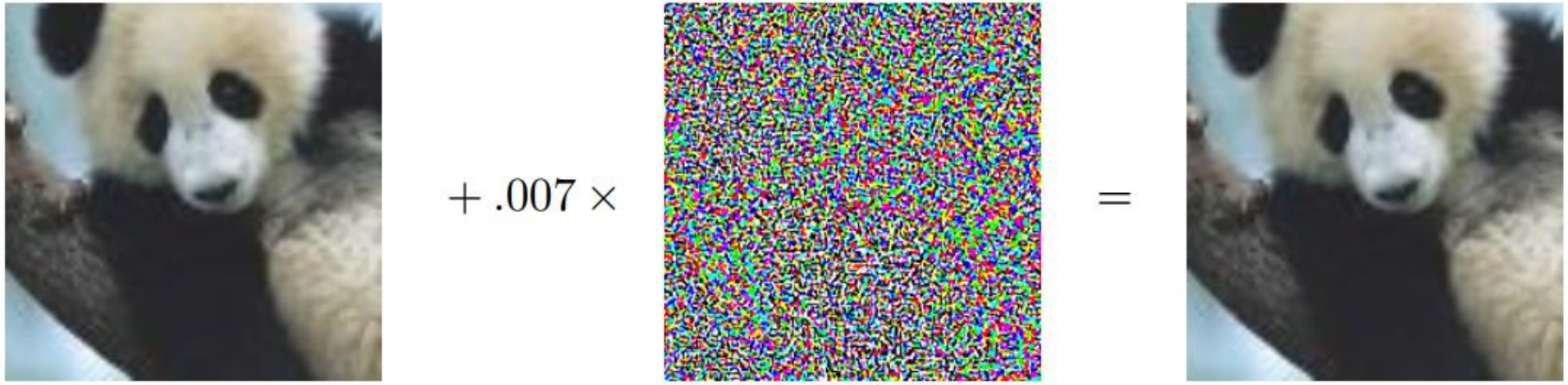
3. We break the unexplored belief that previous detection methods against intentional attacks can capture the valid input range of DNNs

Outline

- Topic 1: Detecting Real-world Corner Cases for DNNs
- **Topic 2: Synthesizing Adversarial Samples against Undefended DNNs**
- Topic 3: Synthesizing Adversarial Samples against Defended DNNs
- Topic 4: Global Explanations of DNNs
- Conclusion and Future Work

Motivation

- Adversarial sample
  - Intentionally crafted inputs that can cause wrong predictions of the models
    - Imperceptible changes to the clean images
    - Unnatural artifacts
  - Intentional failure



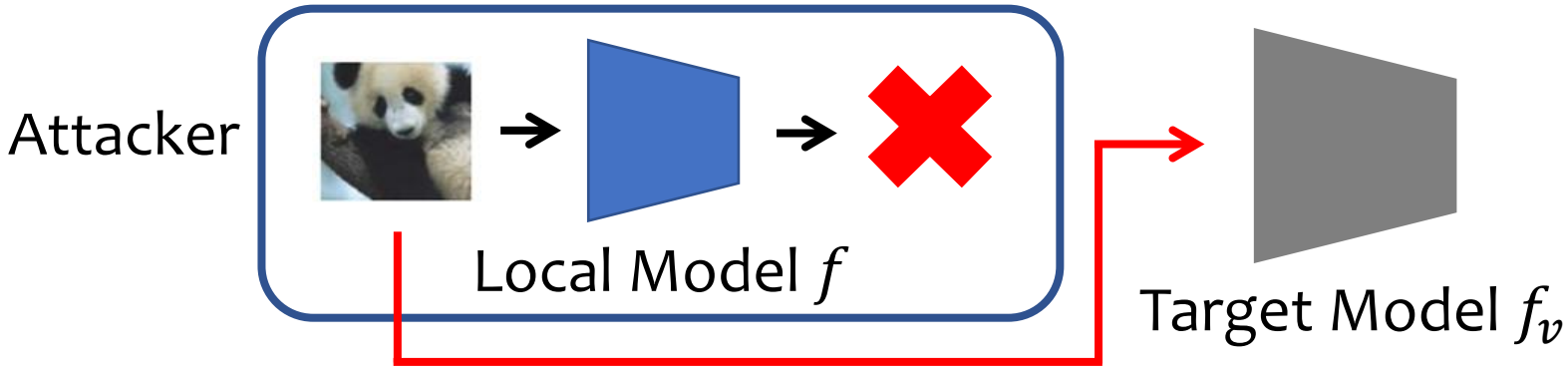
Panda (confidence: 57.7%)

Gibbon (confidence: 99.3%)

[Goodfellow et al., 2015]

Motivation

- Test the robustness of undefended DNNs against intentional failures
  - Attack undefended DNNs by generating adversarial samples under the assumed threat model (in this thesis, the transfer-based setting)
  - The first step to debug
- Transfer-based setting
  - Devise adversarial samples with an off-the-shelf local/source model f
  - Directly use the resultant example to fool the remote target/victim model f_v
  - High threat in practice



Motivation

- Research question
  - How to generate the adversarial counterpart $\mathbf{x}^{adv}$ of a seed image $\mathbf{x}$ under the following transfer-based setting?
    - M: attack method
    - y: ground-truth label
    - ϵ: perturbation budget
    - f_v: undefended victim model, f: undefended local model

$$\mathbf{x}^{adv} = M(f, \mathbf{x})$$

→ Generate an adversarial image $\mathbf{x}^{adv}$ with a local model f by perturbing a seed image $\mathbf{x}$

$$\text{s.t.} \quad \arg\max f_v(\mathbf{x}^{adv}) \neq y$$

→ $\mathbf{x}^{adv}$ is misclassified by the undefended victim model f_v

$$||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$

→ The perturbation is human-imperceptible

Challenge

- Existing effort
  - Employ white-box attack strategies to attack local models
    - J: cross-entropy loss

$$\max_{\mathbf{x}^{adv}} \quad J(f(\mathbf{x}^{adv}), y)$$
$$\text{s.t.} \quad ||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$

→ Maximize the cross-entropy loss of the local model f with respect to the ground-truth label y

- Overfitting issue: low transferability

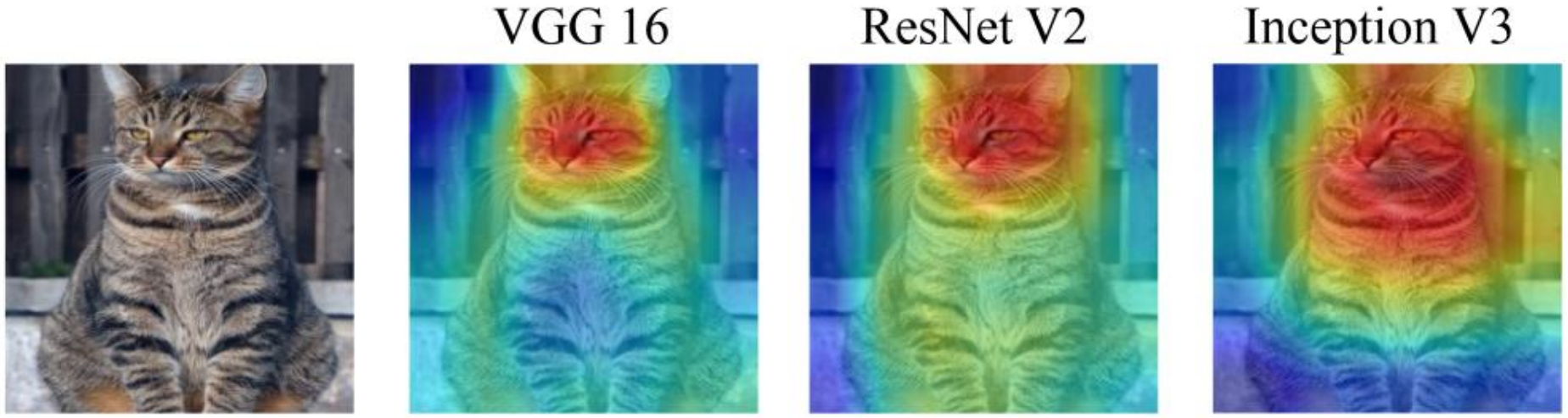
$$\arg\max f(\mathbf{x}^{adv}) \neq y$$ → $\mathbf{x}^{adv}$ can fool the local model f

$$\arg\max f_v(\mathbf{x}^{adv}) = y$$ → $\mathbf{x}^{adv}$ cannot fool the victim model f_v

Method

- Motivation
  - Introduce a regularization term to guide the search of adversarial samples towards the common vulnerable directions of different models

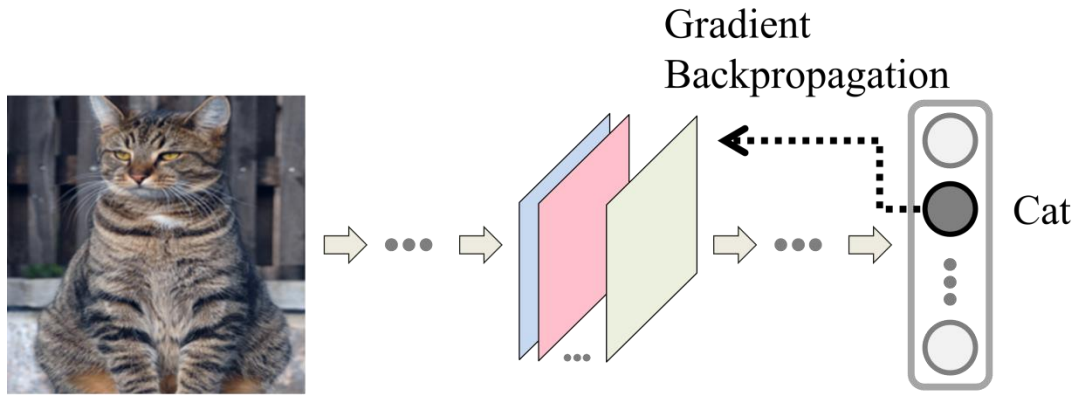
- What different models have in common?
  - Attention pattern: the critical features that models employ to make predictions



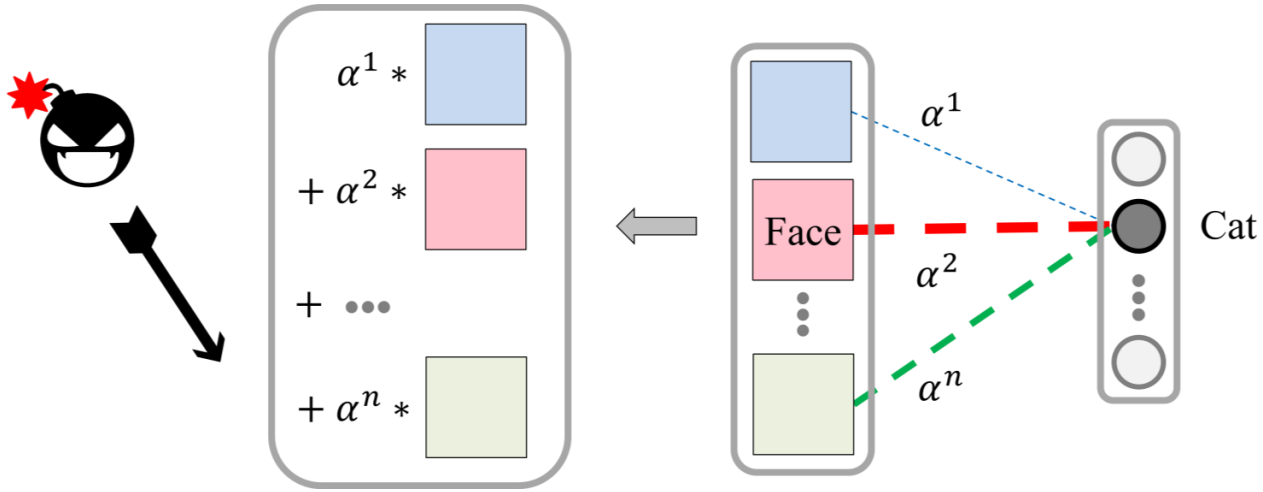
Method

- Attention-guided Transfer Attack (ATA)

Attention Extraction



Attention Destruction

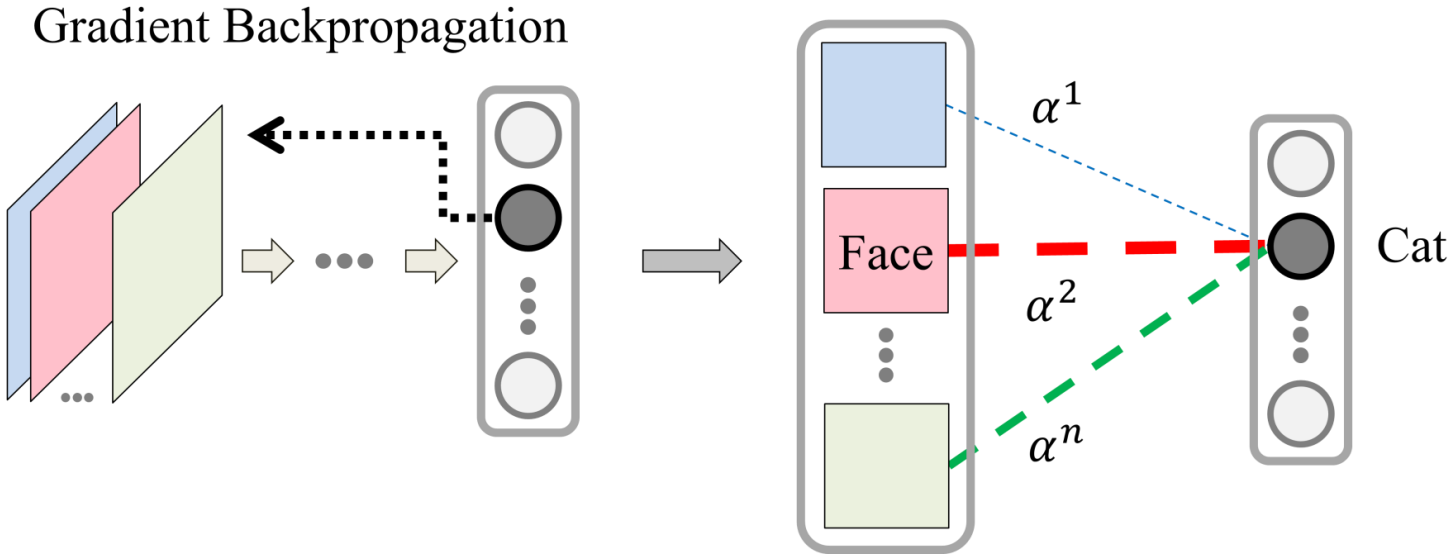


Attention Extraction

- Attention weight
  - A_k^c: the c-th feature map in layer k
  - Z: normalizing constant

$$\alpha_k^c[y] = \frac{1}{Z}\sum_m\sum_n \frac{\partial f(\mathbf{x})[y]}{\partial A_k^c[m,n]}$$

→ Spatially pooled gradients with respect to A_k^c

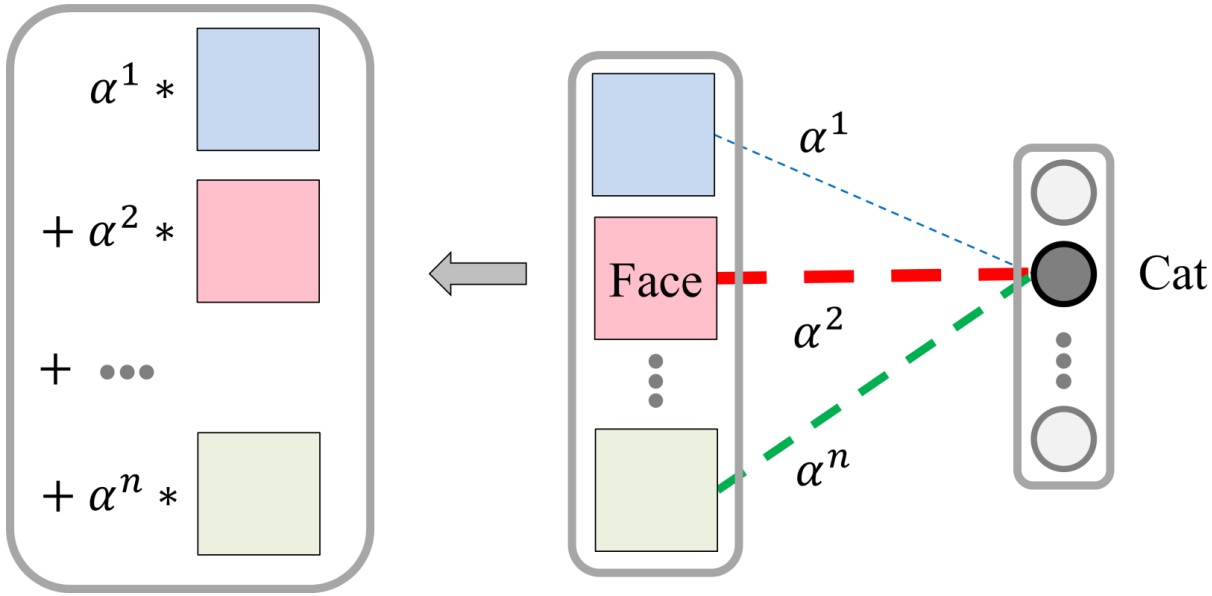


Attention Extraction

- Attention map

$$H_k^y = \mathrm{ReLU}(\sum_c \alpha_k^c[y] \cdot A_k^c)$$

Combine feature maps A_k^c based on their attention weights α_k^c

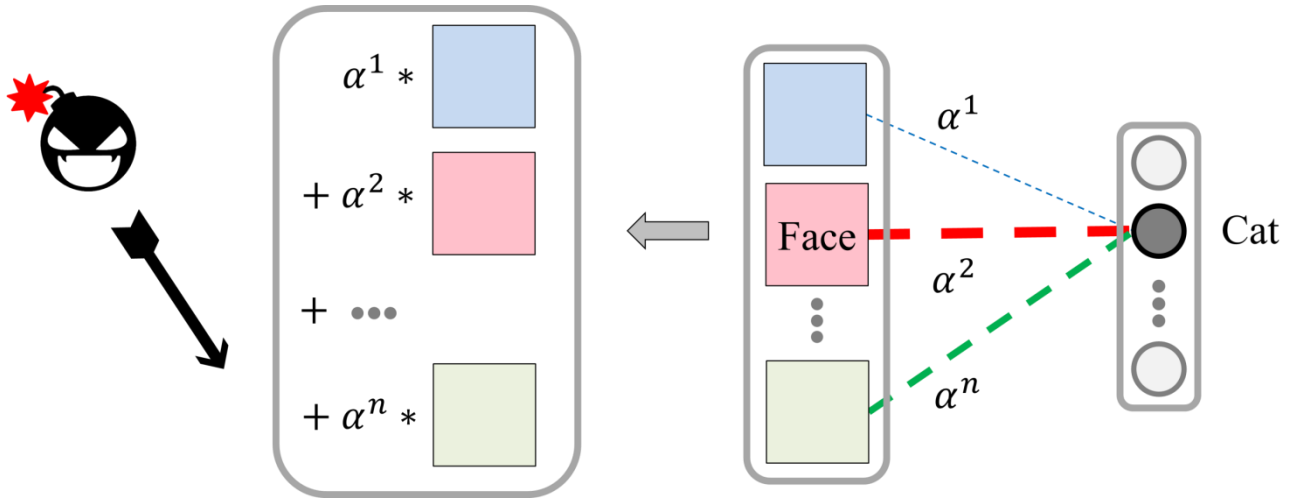


Attention Destruction

- Attack object function
  - The weighted sum of the cross-entropy loss J of the local model f and the changes of the attention maps H_k^y

$$\max_{\mathbf{x}^{adv}} \quad L = J(f(\mathbf{x}^{adv}), y) + \lambda \sum_k ||H_k^y(\mathbf{x}^{adv}) - H_k^y(\mathbf{x})||_2$$

$$\text{s.t.} \quad ||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$



Optimization Algorithm

Algorithm 1 Attention-guided Transfer Attack (ATA)

Require: A classifier f, attack object function L, a clean image $\mathbf{x}$, and its ground-truth label y

Require: The perturbation budget ϵ, iteration number K

Ensure: $||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$

1: $\epsilon' = \dfrac{\epsilon}{K}$

2: $\mathbf{x}_0^{adv} = \mathbf{x}$

3: **for** $k = 0$ to $K - 1$ **do**

4: $\quad \mathbf{x}_{k+1}^{adv} = \text{Clip}_{\mathbf{x}}^{\epsilon}\{\mathbf{x}_k^{adv} + \epsilon' \text{ sign}(\dfrac{\partial L}{\partial \mathbf{x}})\}$

5: **end for**

6: **return** $\mathbf{x}^{adv} = \mathbf{x}_K^{adv}$

Iteratively perturb the current image $\mathbf{x}_k^{adv}$ along with the sign of the gradient of the attack object function

Experiments

- Dataset
  - Development set: ILSVRC 2012 validation set
  - Test set: ImageNet-compatible dataset released by the NeurIPS 2017 adversarial competition
- Baseline
  - White-box attack: FGSM, BIM
  - Transfer-based attack: TAP
- Metric: accuracy on adversarial samples ($\downarrow$)
  - Lower accuracy $\rightarrow$ better attack performance

Experiments

	Attack	Res-v2	Inc-v3	Inc-v4	IncRes-v2	Ensemble
	None	89.60%	96.40%	97.60%	100%	99.80%
Res-v2	FGSM	14.60%	56.30%	64.80%	66.80%	63.10%
	BIM	**4.40%**	53.20%	62.00%	63.80%	54.30%
	TAP	9.50%	**51.20%**	60.10%	55.50%	50.30%
	ATA (Ours)	8.70%	52.90%	**58.30%**	**55.10%**	**49.40%**
Inc-v3	FGSM	65.70%	27.20%	70.20%	72.90%	76.20%
	BIM	76.80%	**0.01%**	67.70%	70.20%	73.60%
	TAP	48.20%	0.10%	24.50%	26.30%	34.20%
	ATA (Ours)	**47.20%**	0.10%	**22.10%**	**25.70%**	**31.90%**
Inc-v4	FGSM	68.30%	67.10%	50.30%	72.80%	76.40%
	BIM	62.10%	40.90%	**0.90%**	69.10%	55.50%
	TAP	**58.40%**	27.30%	1.80%	24.20%	51.70%
	ATA (Ours)	59.90%	**24.80%**	**0.90%**	**22.10%**	**50.30%**
IncRes-v2	FGSM	71.70%	69.00%	76.50%	57.20%	78.70%
	BIM	60.40%	41.50%	51.50%	**1.20%**	54.50%
	TAP	53.30%	25.90%	33.20%	4.80%	48.20%
	ATA (Ours)	**49.80%**	**22.10%**	**30.10%**	**1.20%**	**45.30%**

Experiments

- Sample adversarial image

Clean  Adversarial

- Complementary effect
  - TAP+ATA: add the proposed regularization term to the attack object function of TAP

Attack	Res-v2	Inc-v3	Inc-v4	IncRes-v2	Ensemble
TAP	58.40%	27.30%	1.80%	24.20%	51.70%
TAP+ATA (Ours)	**53.60%**	**22.70%**	**0.80%**	**19.80%**	**48.10%**

Summary

1. We propose a novel Attention-guided Transfer Attack to evaluate the robustness of undefended models against adversarial samples

2. Extensive experiments confirm the effectiveness of our approach and its superiority to state-of-the-art baselines

3. Our strategy can be conveniently combined with other transfer-based attacks to further improve their performance

Outline

- Topic 1: Detecting Real-world Corner Cases for DNNs
- Topic 2: Synthesizing Adversarial Samples against Undefended DNNs
- Topic 3: Synthesizing Adversarial Samples against Defended DNNs
- Topic 4: Global Explanations of DNNs
- Conclusion and Future Work

Motivation

- Test the robustness of defended DNNs against intentional failures
  - Attack defended DNNs by generating adversarial samples under the assumed threat model (in this thesis, the transfer-based setting)
  - Evaluate different defenses
- Research question
  - How to generate the adversarial counterpart $\mathbf{x}^{adv}$ of a seed image $\mathbf{x}$ under the following transfer-based setting?
    - f_{dv}: defended victim model, f: undefended local model

$$\mathbf{x}^{adv} = M(f, \mathbf{x})$$ → Generate an adversarial image $\mathbf{x}^{adv}$ with a local model f by perturbing a seed image $\mathbf{x}$

$$\text{s.t.} \quad \arg\max f_{dv}(\mathbf{x}^{adv}) \neq y$$ → $\mathbf{x}^{adv}$ is misclassified by the defended victim model f_{dv}

$$||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$ → The perturbation is human-imperceptible

Challenge

- Adversarial noise is vulnerable to defenses
  - Overfit to undefended local model
  - Small magnitude: easy to "de-noise" adversarial samples via image transformations (transformation-based defenses)

- Existing effort
  - Data augmentation: train adversarial samples to become effective against common image transformations, like resizing
  - Drawback: overfit to the applied image transformations

Method

- Motivation
  - Augment the hardest transformations (data)

- Challenge
  - How to identify the most harmful image transformation to an adversarial image $\mathbf{x}^{adv}$?
    - H: image transformation function with the parameter θ_H
    - Inner maximization: find the adversarial image $\mathbf{x}^{adv}$ to cause misclassification
    - Outer minimization: find the image transformation H to de-noise $\mathbf{x}^{adv}$

$$\min_{\theta_H} \max_{\mathbf{x}^{adv}} \quad J(f(H(\mathbf{x}^{adv})), y)$$

$$\text{s.t.} \quad ||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$

$$\arg\max \; f(H(\mathbf{x})) = y \longrightarrow$$

The image transformation H itself will not cause misclassification

Method

- Workaround
  - Restrict the hypothesis space of *H* to be some class of convolutional neural networks $T(\mathbf{x};\ \theta_T)$ – adversarial transformation network

$$\min_{\theta_H} \max_{\mathbf{x}^{adv}} \quad J(f(H(\mathbf{x}^{adv})), y)$$
$$\text{s.t.} \quad ||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$
$$\arg\max \ f(H(\mathbf{x})) = y$$

$$\min_{\theta_T} \max_{\mathbf{x}^{adv}} \quad J(f(T(\mathbf{x}^{adv})), y)$$
$$\text{s.t.} \quad ||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$
$$\arg\max \ f(T(\mathbf{x})) = y$$

- Merit
  - CNNs possess the capacity to generate diverse image distortions
  - Convenient to learn T in an end-to-end manner

Method

- How to solve the min-max problem?

$$\min_{\theta_T} \max_{\mathbf{x}^{adv}} \quad J(f(T(\mathbf{x}^{adv})), y)$$
$$\text{s.t.} \quad ||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$
$$\arg\max \; f(T(\mathbf{x})) = y$$

- Outer minimization
  - Training loss function of the adversarial transformation network T
    - ①: minimize the cross-entropy loss J on the transformed adversarial image $T(\mathbf{x}^{adv})$
    - ②: minimize the cross-entropy loss J on the transformed clean image $T(\mathbf{x})$
    - ③: control the transformation strength to perform regularization

$$L_T = \underbrace{J(f(T(\mathbf{x}^{adv})), y)}_{①} + \alpha_1 \underbrace{J(f(T(\mathbf{x})), y)}_{②} + \alpha_2 \underbrace{||\mathbf{x}^{adv} - T(\mathbf{x}^{adv})||^2}_{③}$$

Method

- How to solve the min-max problem?

$$\min_{\theta_T} \max_{\mathbf{x}^{adv}} \quad J(f(T(\mathbf{x}^{adv})), y)$$
$$\text{s.t.} \quad ||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$$
$$\arg\max \; f(T(\mathbf{x})) = y$$

- Inner maximization
  - Fooling object function to search for the adversarial image $\mathbf{x}^{adv}$
    - ①: maximize the cross-entropy loss J on the transformed adversarial image $T(\mathbf{x}^{adv})$
    - ②: maximize the cross-entropy loss J on the adversarial image $\mathbf{x}^{adv}$

$$L_{fool} = -J(f(T(\mathbf{x}^{adv})), y) - \beta J(f(\mathbf{x}^{adv}), y)$$
$$\text{①} \qquad\qquad \text{②}$$

Training Algorithm

Algorithm 1 Adversarial Transformation Network Training

Require: The fooling object function L_{fool}, the training loss function L_T of the adversarial transformation network, and a clean image $\mathbf{x}$

Require: The perturbation budget ϵ, the iteration numbers K_{outer} and K_{inner}

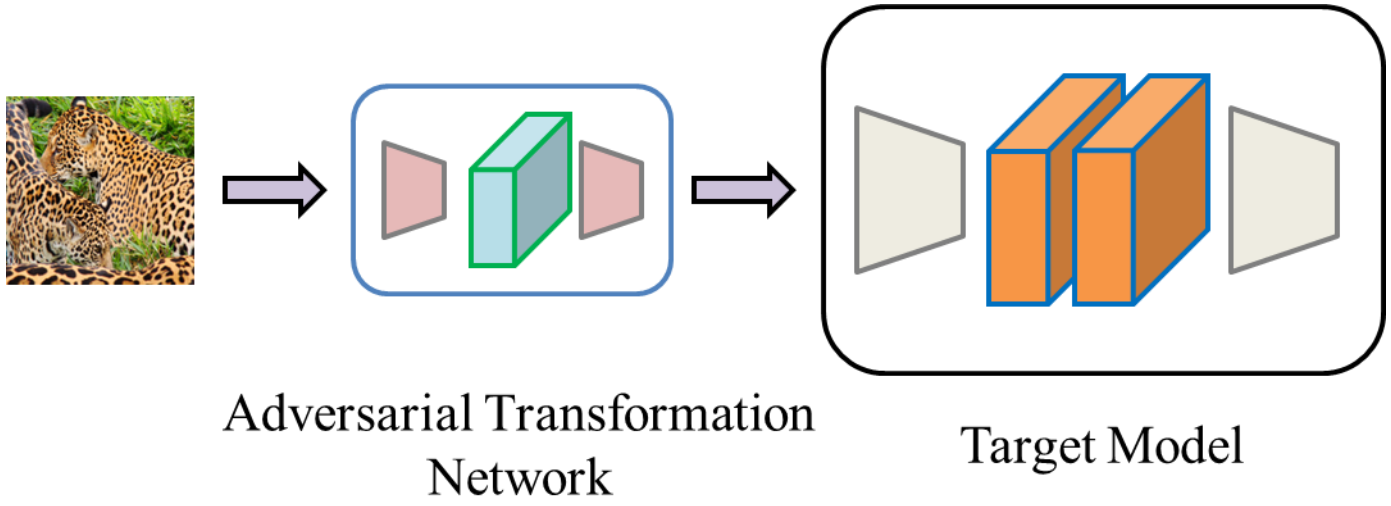
1: Initialize $\mathbf{x}^{adv} = \mathbf{x}$
2: Randomly initialize θ_T
3: **for** $k_{outer} = 1$ to K_{outer} **do**
4: **for** $k_{inner} = 1$ to K_{inner} **do**
5: Update $\mathbf{x}^{adv} = \text{Clip}_{\mathbf{x}}^{\epsilon}\{\mathbf{x}^{adv} - \text{Adam}(L_{fool})\}$ ➡ Iteratively solve the inner maximization problem
6: **end for**
7: Update $\theta_T = \theta_T - \text{Adam}(L_T)$ ➡ Iteratively solve the outer minimization problem
8: **end for**
9: **return** the parameter θ_T of the learned adversarial transformation network

Method

- Adversarial Transformation-enhanced Transfer Attack (ATTA)
  - Attack object function
    - ①: maximize the cross-entropy loss J on the adversarial sample $\mathbf{x}^{adv}$
    - ②: maximize the cross-entropy loss J on the transformed adversarial sample $T(\mathbf{x}^{adv})$

$$L_{attack} = \underbrace{J(f(\mathbf{x}^{adv}), y)}_{①} + \gamma \underbrace{J(f(T(\mathbf{x}^{adv})), y)}_{②}$$

The Cascaded Network
$f(T(\mathbf{x}))$



Optimization Algorithm

Algorithm 2 Adversarial Sample Generation

Require: A classifier f, the attack object function L_{attack}, the adversarial transformation network T, a clean image $\mathbf{x}$, and its ground-truth label y

Require: The perturbation budget ϵ and iteration number K

Ensure: $||\mathbf{x}^{adv} - \mathbf{x}||_\infty \leq \epsilon$

1: $\epsilon' = \dfrac{\epsilon}{K}$

2: $\mathbf{x}_0^{adv} = \mathbf{x}$

3: **for** $k = 0$ to $K - 1$ **do**

4: $\mathbf{x}_{k+1}^{adv} = \mathrm{Clip}_{\mathbf{x}}^{\epsilon}\{\mathbf{x}_k^{adv} + \epsilon' \,\mathrm{sign}(\dfrac{\partial L_{attack}}{\partial \mathbf{x}})\}$

5: **end for**

6: **return** $\mathbf{x}^{adv} = \mathbf{x}_K^{adv}$

Iteratively perturb the current image $\mathbf{x}_k^{adv}$ along with the sign of the gradient of the attack object function

Experiments

- Dataset
  - Development set: ILSVRC 2012 training set
  - Test set: ILSVRC 2012 validation set
- Target model
  - Defended model: adversarial training, transformation-based defense
  - Undefended model
- Baseline
  - White-box attack: FGSM, BIM
  - Transfer-based attack: TIM
- Metric: fooling rate (↑)
  - Error rate on adversarial samples
  - Higher fooling rate → better attack performance

Experiments

- Attack undefended and adversarially trained models

	Attack	Res-v2	Inc-v3	Inc-v4	IncRes-v2	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{adv}$
Res-v2	FGSM	85.4	43.7	35.2	33.2	22.6	22.2	14.3
	BIM	95.6	46.8	38.0	36.2	27.6	25.3	17.4
	TIM	98.8	**65.2**	59.8	57.4	35.6	31.7	25.8
	ATTA (Ours)	**99.8**	64.3	**61.8**	**59.2**	**42.1**	**38.9**	**29.1**
Inc-v3	FGSM	34.3	72.8	29.8	27.1	14.9	13.6	17.9
	BIM	33.2	99.9	32.3	29.8	11.8	11.5	17.6
	TIM	39.2	**100**	44.3	45.8	23.2	24.9	16.4
	ATTA (Ours)	**44.8**	**100**	**52.9**	**53.2**	**25.1**	**27.9**	**18.8**
Inc-v4	FGSM	31.7	32.9	49.7	28.2	11.9	13.1	6.2
	BIM	37.9	59.1	99.1	30.9	14.7	14.7	7.1
	TIM	41.4	64.3	**99.6**	48.2	25.7	25.2	16.9
	ATTA (Ours)	**43.8**	**66.8**	**99.6**	**59.2**	**32.1**	**29.2**	**20.8**
IncRes-v2	FGSM	29.3	31.0	23.5	42.8	13.1	12.7	7.3
	BIM	39.6	58.5	23.5	42.8	15.2	13.1	7.1
	TIM	43.1	62.9	55.4	**98.9**	31.8	29.2	20.6
	ATTA (Ours)	**44.8**	**68.9**	**65.2**	**98.9**	**33.0**	**31.9**	**24.3**

Experiments

- Attack transformation-based defenses

Attack	HGD	R&P	NIPS-r3	FD	ComDefend	RS	Average
FGSM	8.9	16.8	23.1	19.2	13.4	6.8	14.7
BIM	12.1	19.3	23.8	21.8	17.2	8.9	17.2
TIM	73.3	69.8	79.4	78.2	69.2	36.2	67.7
ATTA (Ours)	**85.9**	**83.2**	**89.5**	**84.4**	**79.9**	**47.4**	**78.4**

- Sample adversarial image

Clean Transformed Adversarial

Experiments

- Complementary effect
  - Easy to combine our method with others
  - Attack both the original classifier and the network cascaded with T via SI-NI-TI-DIM

	Attack	Res-v2	Inc-v3	Inc-v4	IncRes-v2	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{adv}$
Res-v2	SI-NI-TI-DIM	**99.8**	78.3	70.2	71.8	34.9	35.9	30.2
	AT-SI-NI-TI-DIM (Ours)	**99.8**	**80.1**	**74.9**	**74.9**	**36.8**	**37.3**	**33.2**
Inc-v3	SI-NI-TI-DIM	48.3	**100**	54.3	56.2	**27.8**	28.1	24.5
	AT-SI-NI-TI-DIM (Ours)	**49.1**	**100**	**55.9**	**57.1**	**27.8**	**28.6**	**24.9**
Inc-v4	SI-NI-TI-DIM	49.5	72.1	**99.6**	60.3	33.2	31.8	26.9
	AT-SI-NI-TI-DIM (Ours)	**50.4**	**75.2**	**99.6**	**62.8**	**33.9**	**32.3**	**27.6**
IncRes-v2	SI-NI-TI-DIM	50.1	72.9	69.6	**98.9**	34.5	32.7	27.4
	AT-SI-NI-TI-DIM (Ours)	**55.3**	**77.8**	**74.2**	**98.9**	**36.5**	**34.9**	**29.1**

Summary

1. We propose a novel Adversarial Transformation-enhanced Transfer Attack to evaluate the robustness of defended models against adversarial samples

2. Extensive experiments confirm the effectiveness of our approach and its superiority to state-of-the-art baselines

3. Our strategy can be conveniently combined with other transfer-based attacks to further improve their performance

Outline

- Topic 1: Detecting Real-world Corner Cases for DNNs
- Topic 2: Synthesizing Adversarial Samples against Undefended DNNs
- Topic 3: Synthesizing Adversarial Samples against Defended DNNs
- Topic 4: Global Explanations of DNNs
- Conclusion and Future Work

Motivation

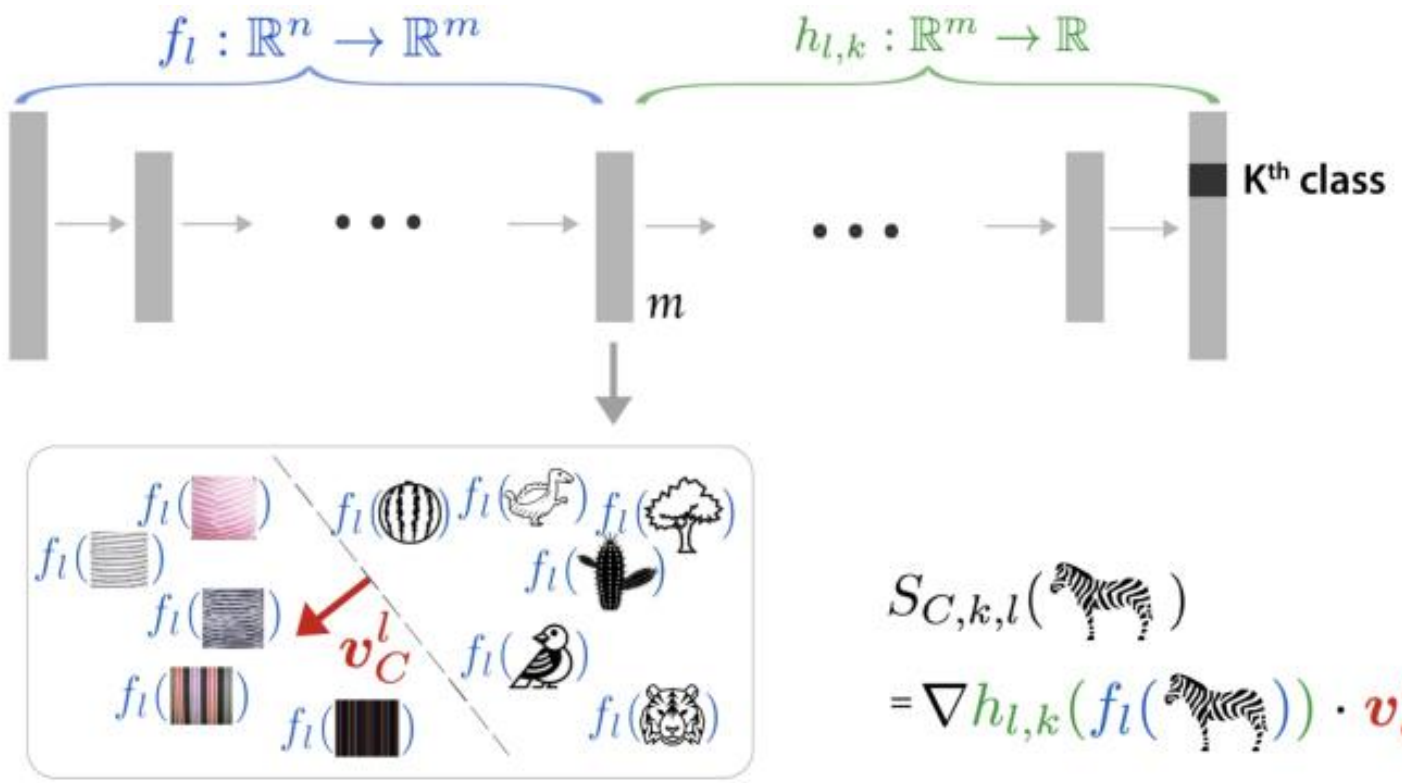
- Explain and understand the behaviors of DNNs
  - Promote the interpretability and trustworthiness of DNNs: prerequisite for the broad deployment of DNNs
  - Spot latent defects, e.g., robustness issues

- Research question
  - How to obtain global explanations of DNNs?
    - Global: category-wide
    - Explanation: concept attribution

- Concept attribution
  - Measure the importance of human-understandable notions to model predictions
    - E.g., to what extent the banded texture is related to the prediction of a zebra
  - Merit
    - Directly bridge the discrepant thinking of humans and models

Challenge

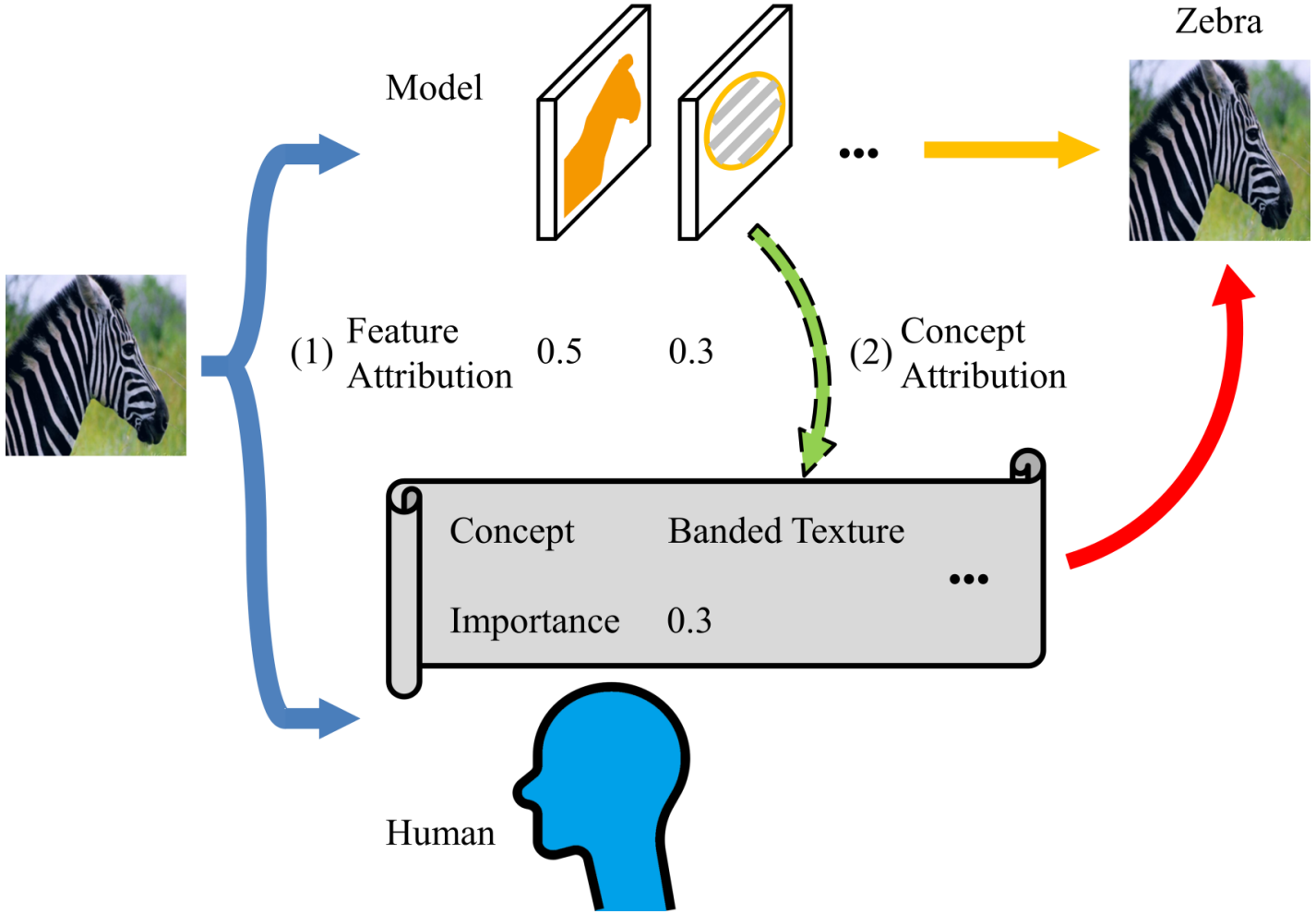
- Existing effort
  - Only consider the proximity of individual instances
  - Drawback: myopic conclusion



[Kim et al., 2018]

Method

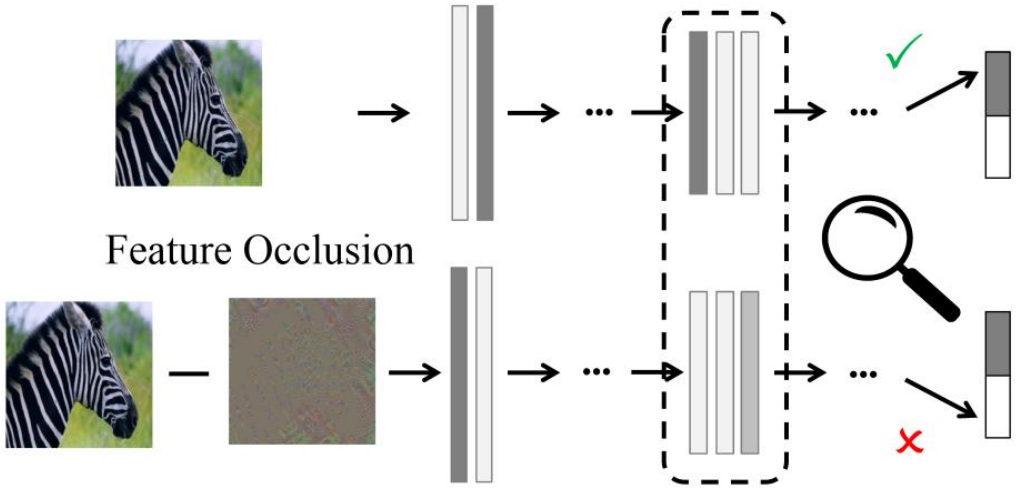
- Motivation

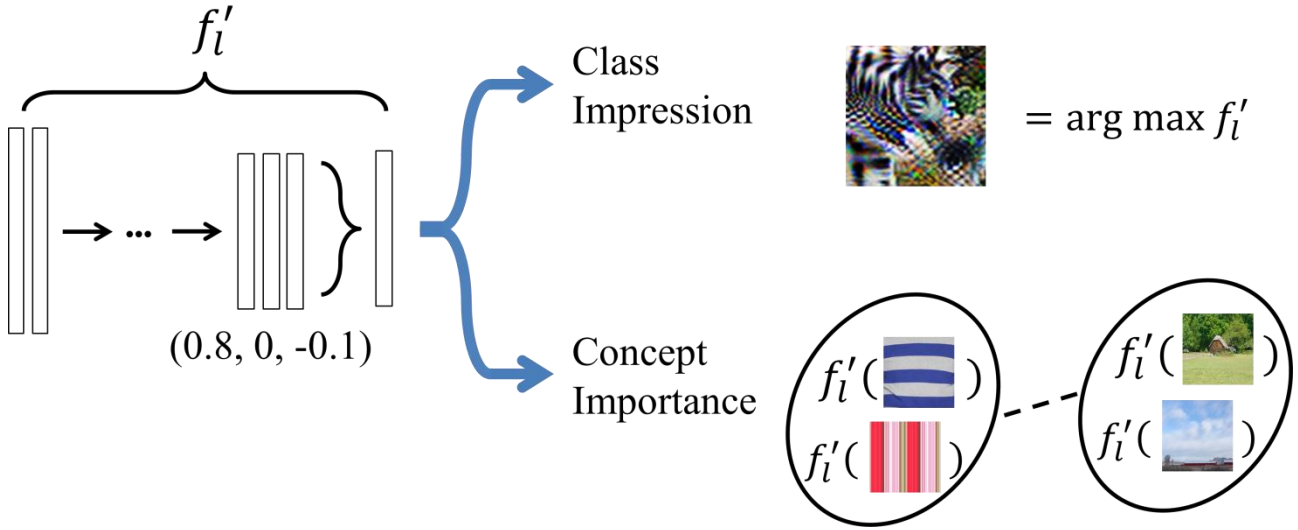


(1) Feature attribution: feature importance

(2) Concept attribution: concept importance

Attacking for Interpretability (AfI)





(1) Feature attribution: feature occlusion

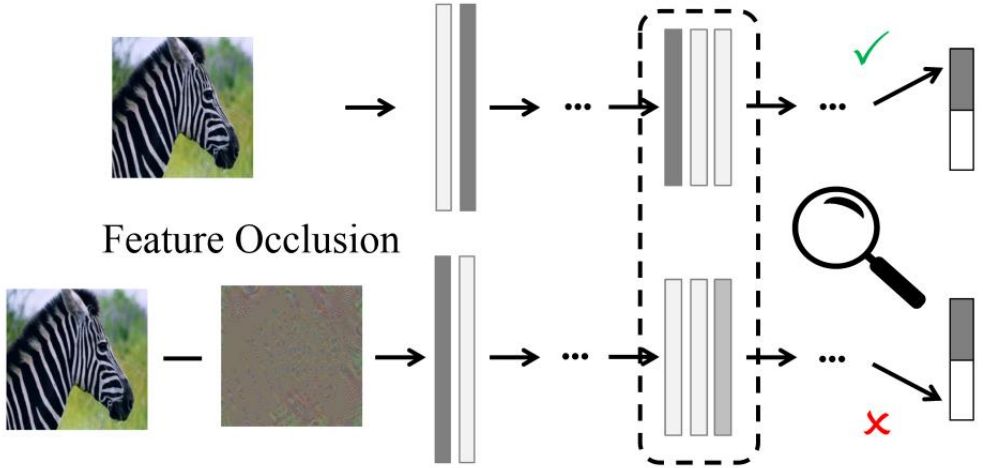
(2) Concept attribution: semantic task

Feature Occlusion

- Motivation
  - The basic reasoning process of a model
    - The features of class y in image $\mathbf{x}$ are more prominent $\Leftrightarrow$ the label prediction for image $\mathbf{x}$ is y
  - Logic equivalence
    - (1) The label prediction for image $\mathbf{x}$ **is** y $\rightarrow$

    the features of class y in image $\mathbf{x}$ are **more** prominent
    - (2) The label prediction for image $\mathbf{x}$ **is not** y $\rightarrow$

    the features of class y in image $\mathbf{x}$ are **less** prominent



Feature Occlusion

- Learn a global feature occluder δ^* to perform feature occlusion (attacking)
  - D: magnitude measure (l_1 norm)
  - t: transformation function – (1) applying uniform random noise and (2) random rotation

$$\delta^* = \arg\min\ D(\delta)$$ → Find the minimal occluder δ^*

$$\text{s.t.}\quad f(\mathbf{x}_i - \delta) \neq y$$ → δ can cause misclassification

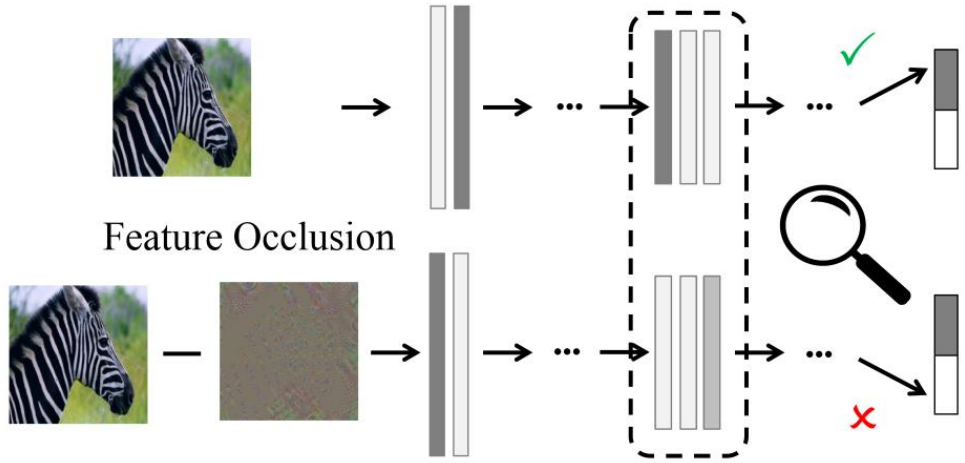
$$f(t(\mathbf{x}_i - \delta)) \neq y$$ → δ can cause misclassification even after the transformation t

$$f(t(\mathbf{x}_i)) = f(\mathbf{x}_i) = y$$ → The original image $\mathbf{x}_i$ and the transformed one $t(\mathbf{x}_i)$ are correctly classified

$$\text{for}\quad i = 1, \ldots, N$$ → δ^* works for a class of samples

Feature Attribution

- Compute feature importance $s_l'^j$
  - The importance score of the feature that the j-th neuron in the l-th layer detects
    - (1) The label prediction for image **x is** y → the features of class y in image **x** are **more** prominent
    - (2) The label prediction for image **x is not** y → the features of class y in image **x** are **less** prominent



The average change of the neuron's outputs after occlusion over a class of samples

$$s_l^j = \frac{1}{N}\Sigma_{i=1}^{N}(f_l(\mathbf{x}_i)[j] - f_l(\mathbf{x}_i - \delta^*)[j])$$

$$s_l'^j = \max(s_l^j, 0)$$

Remove negative importance

Concept Attribution

- Derive a class-specific meta-detector f_l'
  - Channel importance score (CIS)
    - B: normalizing constant
    - P_l^c: the index set of neurons in the c-th feature map of layer l

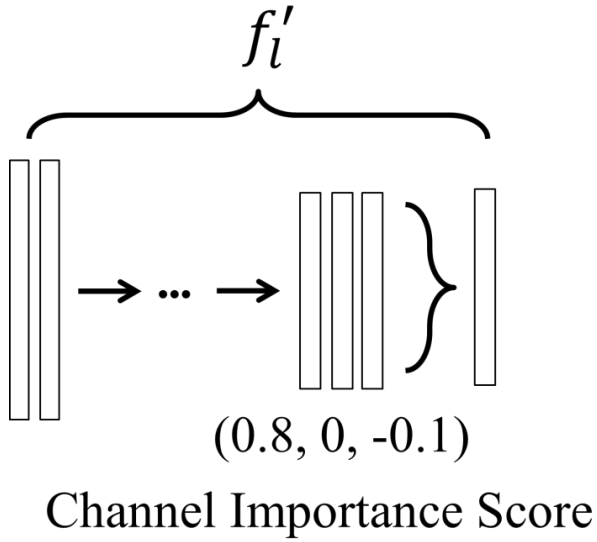
$$w_l^c = \frac{1}{B}\Sigma_{j \in P_l^c} s_l^{'j}$$

→ Spatially pooled feature importance scores $s_l^{'j}$ of a feature map

  - Meta-detector
    - A_l^c: the c-th feature map of layer l

$$f_l^{'} = \Sigma_c w_l^c \cdot A_l^c$$

→ Combine feature maps A_l^c based on channel importance scores ω_l^c

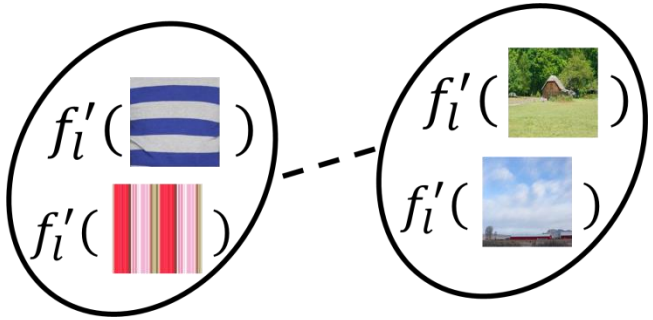


Concept Attribution

- Concept importance: the representation capacity of the meta-detector for a concept of interest
  - Qualitative attribution: generation task
    - Visualize the class concept
    - Generate images that can highly activate the meta-detector

$= \arg\max f_l'$

  - Quantitative attribution: concept classification
    - Measure the importance of user-defined concepts
    - Discrepancy of the concept data to random ones: Maximum Mean Discrepancy (MMD) as the measure



Experiments

- Dataset
  - ImageNet (ILSVRC2012) : ResNet-50, GoogLeNet, VGG-16
- Baseline
  - TCAV
- Metric
  - The smallest sufficient concepts (SSCs): the smallest set of concepts sufficing for models to predict the target class
  - The smallest destroying concepts (SDCs): the smallest concept collections whose absence will incur wrong predictions
  - More accurate estimations of SSCs and SDCs → more accurate estimations of concept importance

Experiments

- Evaluation of the concept attribution results
  - Regard semantic image segments as the representation of concepts

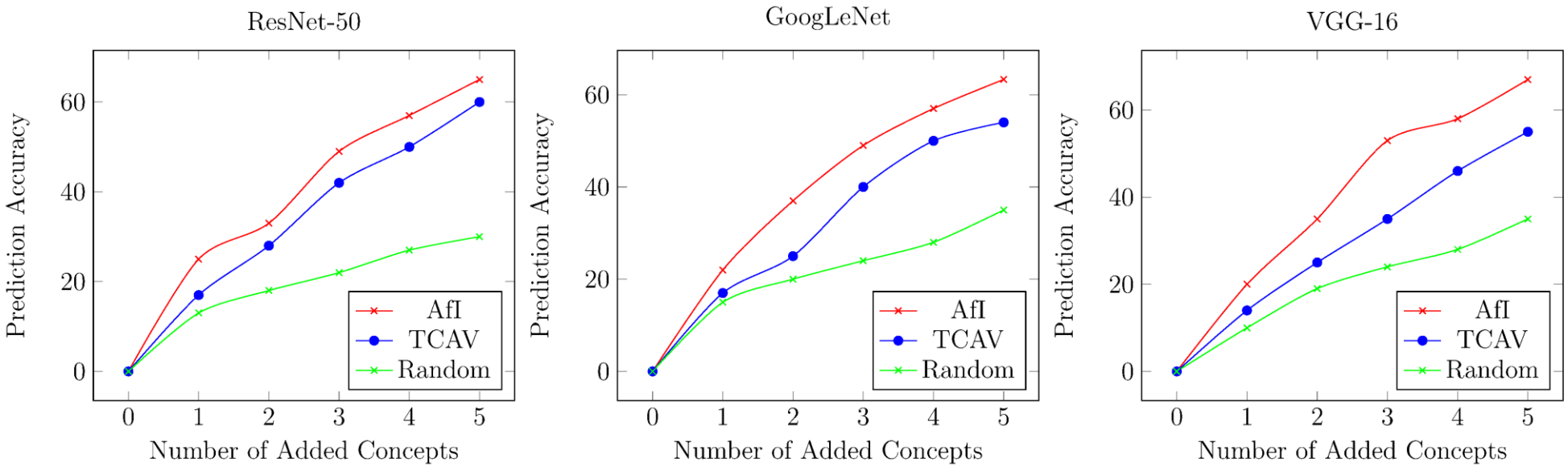
SSC

SDC

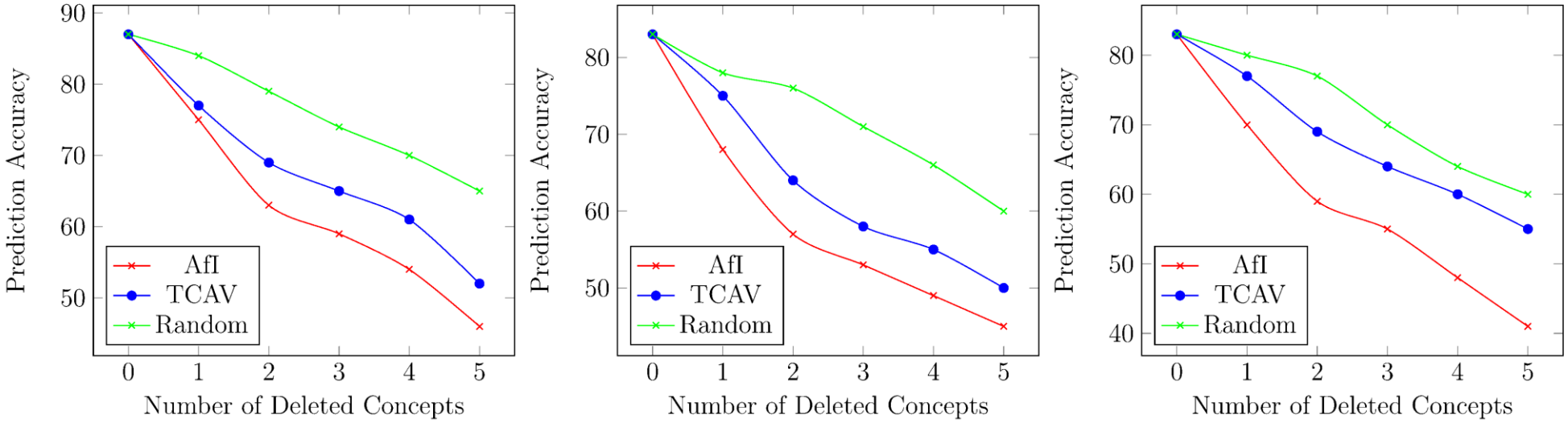
Top - 1 Top - 5 Top - 10

Experiments

SSC

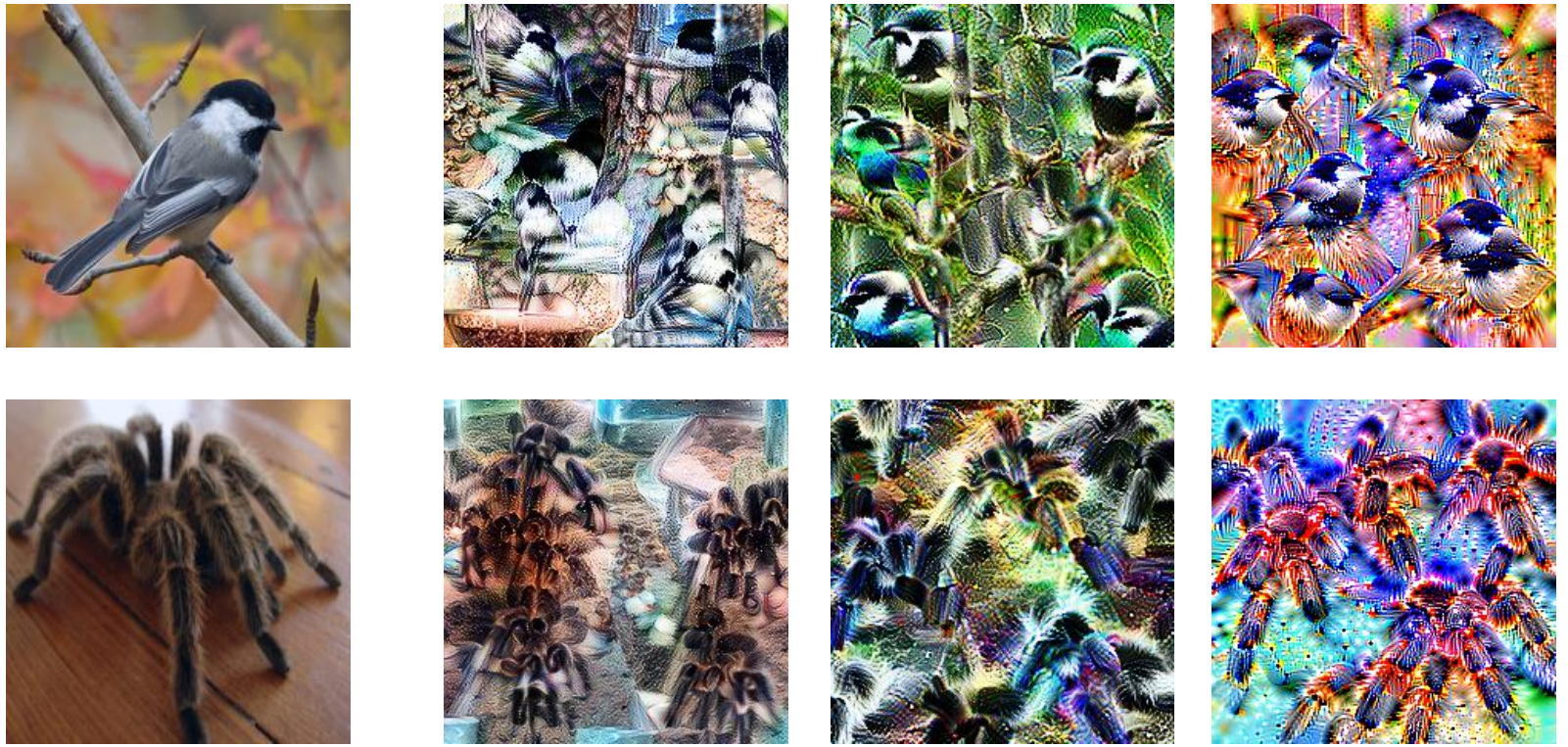


SDC



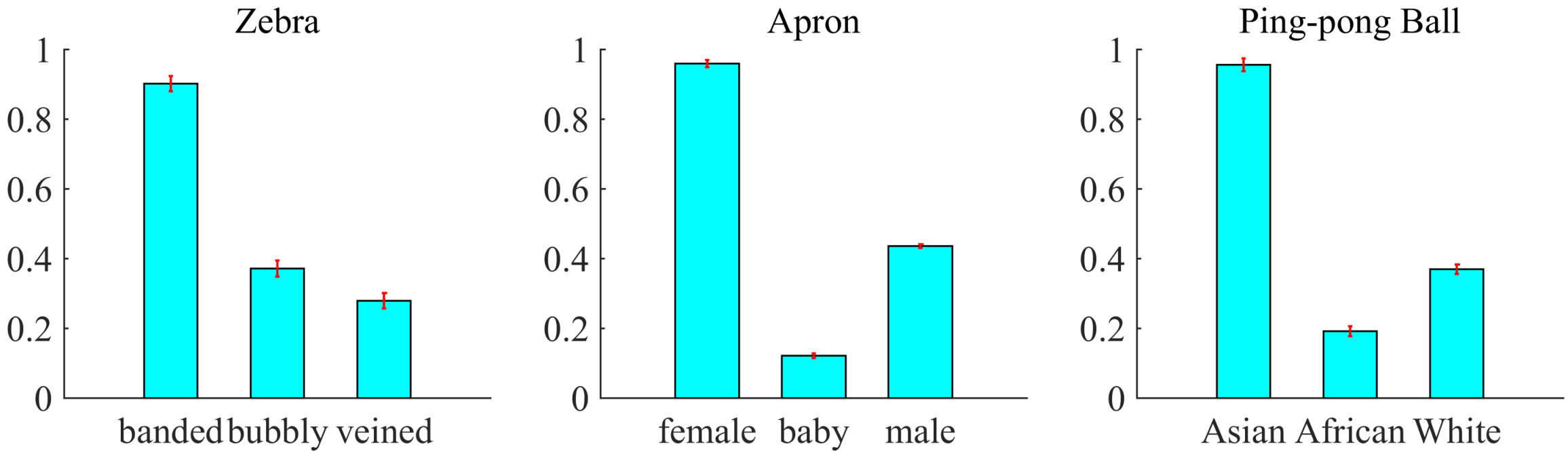
Experiments

- Class concept visualization



Experiments

- User-defined concept attribution



Summary

1. We propose a novel concept attribution framework (Attacking for Interpretability) for global explanations of DNNs

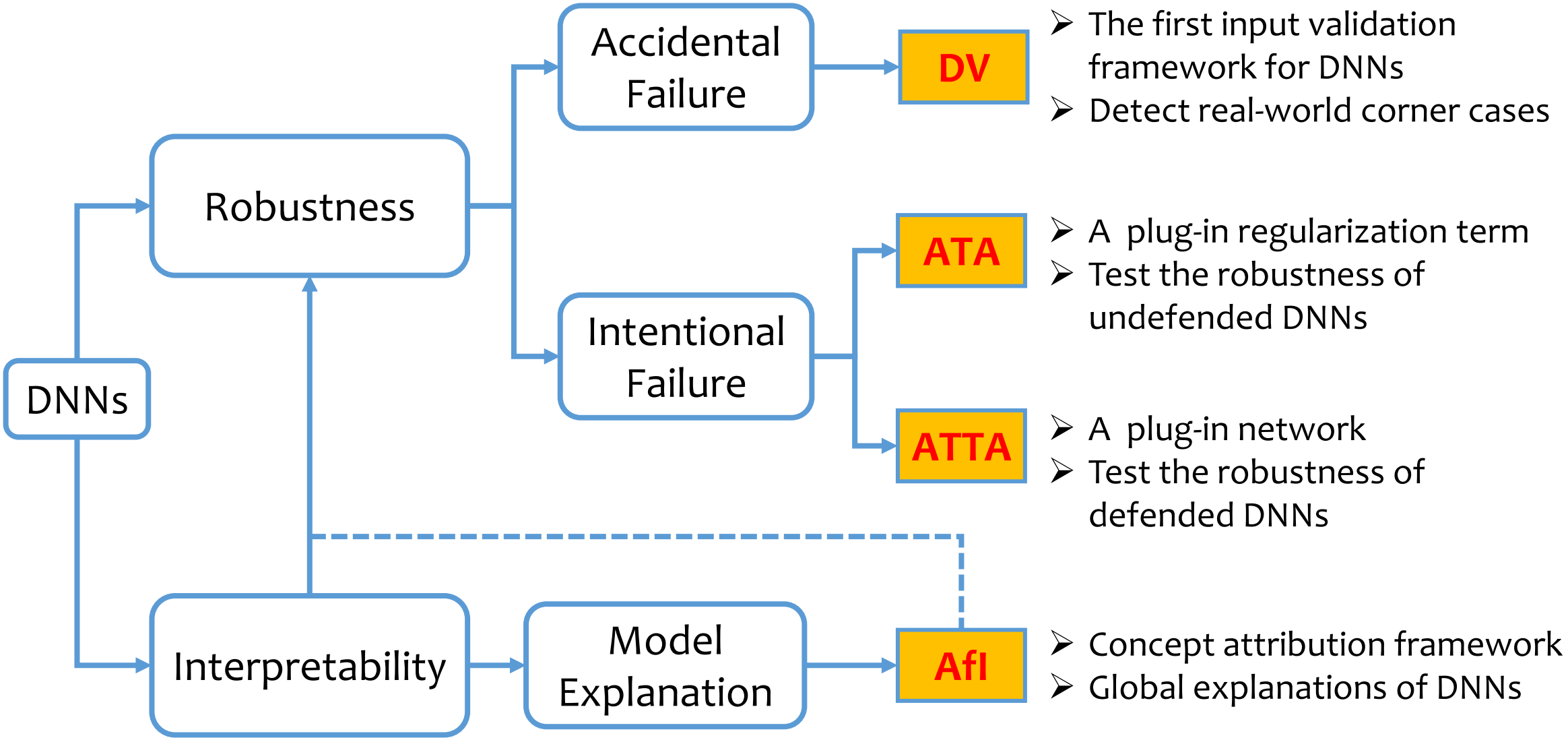
2. Experimental results show that our framework provides more accurate estimations of concept importance than existing proposals

3. We demonstrate the use cases of our method in providing insights into DNNs

Outline

- Topic 1: Detecting Real-world Corner Cases for DNNs
- Topic 2: Synthesizing Adversarial Samples against Undefended DNNs
- Topic 3: Synthesizing Adversarial Samples against Defended DNNs
- Topic 4: Global Explanations of DNNs
- Conclusion and Future Work

Conclusion



- The first input validation framework for DNNs
- Detect real-world corner cases

- A plug-in regularization term
- Test the robustness of undefended DNNs

- A plug-in network
- Test the robustness of defended DNNs

- Concept attribution framework
- Global explanations of DNNs

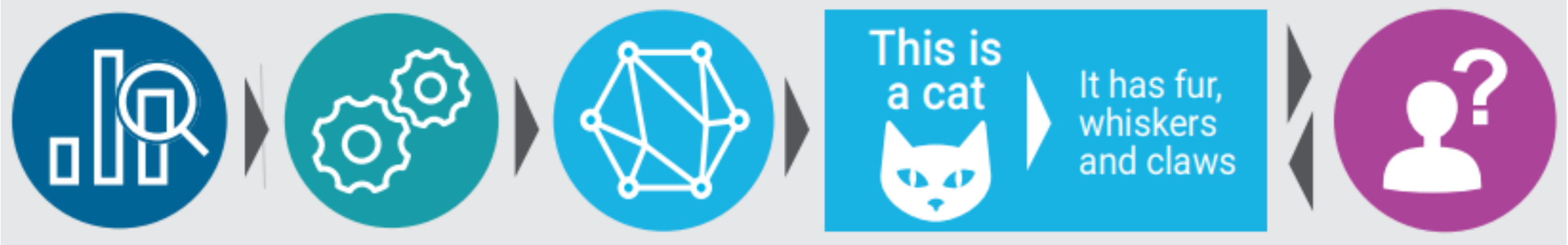
Future Work

- Test the robustness of DNNs against accidental failures
  - Synthesize diverse real-world corner cases
  - Challenge
    - Existing image transformation techniques have limited diversity
    - Test oracle problem

[Pei et al., 2017]

Future Work

- Self-explainable DNNs
  - Produce both decisions and explanations
  - Challenge
    - Require a new learning paradigm



Training Data | New Learning Process | Model | Output | User

Publications

1. **Weibin Wu**, Yuxin Su, Michael R. Lyu, and Irwin King. Improving the Transferability of Adversarial Samples with Adversarial Transformations. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021.
2. **Weibin Wu**, Yuxin Su, Xixian Chen, Shenglin Zhao, Irwin King, Michael R. Lyu, and Yu-Wing Tai. Towards Global Explanations of Convolutional Neural Networks with Concept Attribution (**Oral Presentation**). IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
3. **Weibin Wu**, Yuxin Su, Xixian Chen, Shenglin Zhao, Irwin King, Michael R. Lyu, and Yu-Wing Tai. Boosting the Transferability of Adversarial Samples via Attention. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
4. **Weibin Wu**, Hui Xu, Sanqiang Zhong, Michael R. Lyu, and Irwin King. Deep Validation: Toward Detecting Real-world Corner Cases for Deep Neural Networks. 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2019.
5. Hui Xu, Zhuangbin Chen, **Weibin Wu**, Zhi Jin, Sy-Yen Kuo, and Michael R. Lyu. NV-DNN: Towards Fault-Tolerant DNN Systems with N-Version Programming. 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), 2019.

Thanks!

References

- IEEE Standards Coordinating Committee. (1990). IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990). Los Alamitos. CA: IEEE Computer Society, 169, 132.
- Molnar, C. (2020). Interpretable Machine Learning. Lulu. com.
- Zhang, M., Zhang, Y., Zhang, L., Liu, C., & Khurshid, S. (2018). DeepRoad: GAN-based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems. In the 33rd ACM/IEEE International Conference on Automated Software Engineering (pp. 132-142).
- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. In International Conference on Learning Representations.
- Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., & Viegas, F. (2018). Interpretability beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In International Conference on Machine Learning (pp. 2668-2677).
- Pei, K., Cao, Y., Yang, J., & Jana, S. (2017). Towards Practical Verification of Machine Learning: The Case of Computer Vision Systems. arXiv preprint arXiv:1712.01785.