

**Spring 2021**

# ADVANCED TOPICS IN COMPUTER VISION

---

**Atlas Wang**

Assistant Professor, The University of Texas at Austin

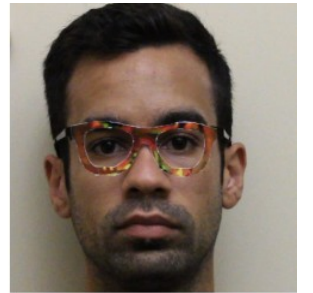
**Visual Informatics Group@UT Austin**

<https://vita-group.github.io/>

# Trustworthy Computer Vision?



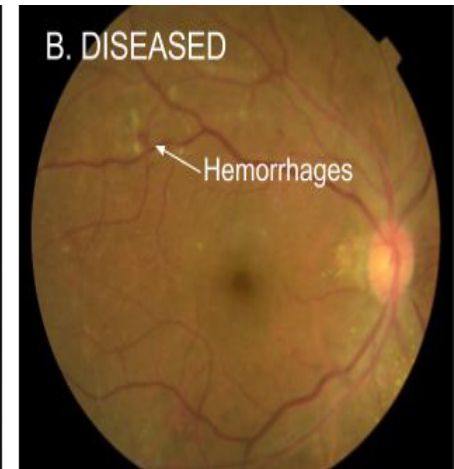
Self-Driving Perception



Face  
Recognition



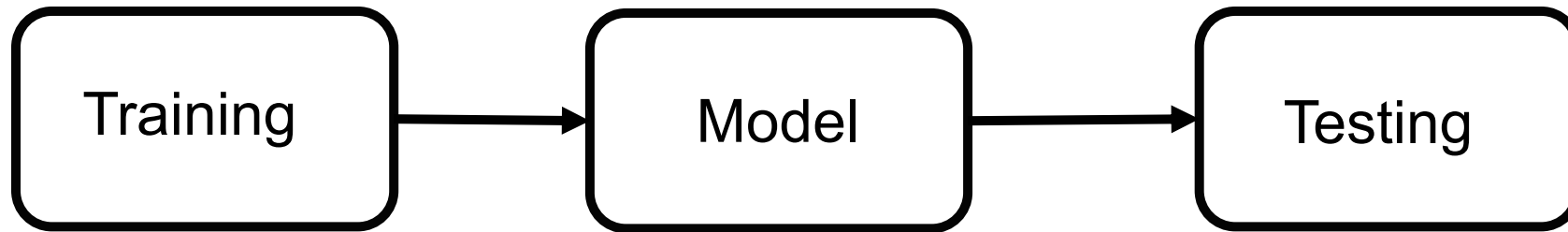
Safe Control



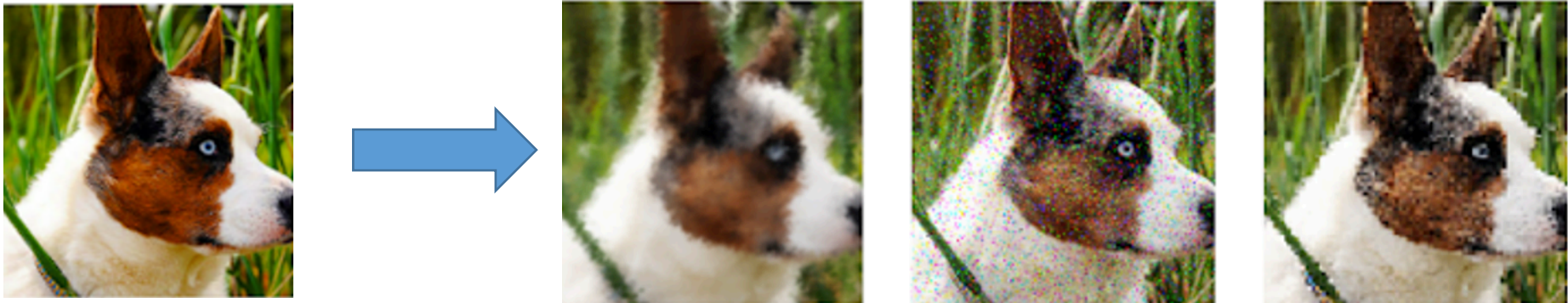
Medical Diagnosis

# Failure Mode I: Data Violate Assumptions

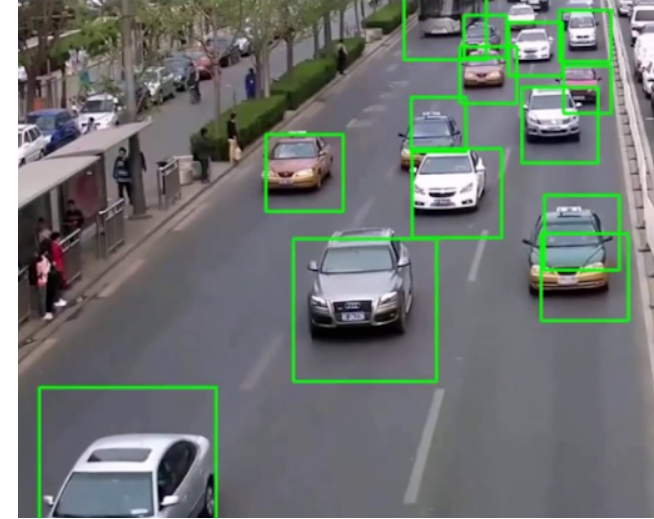
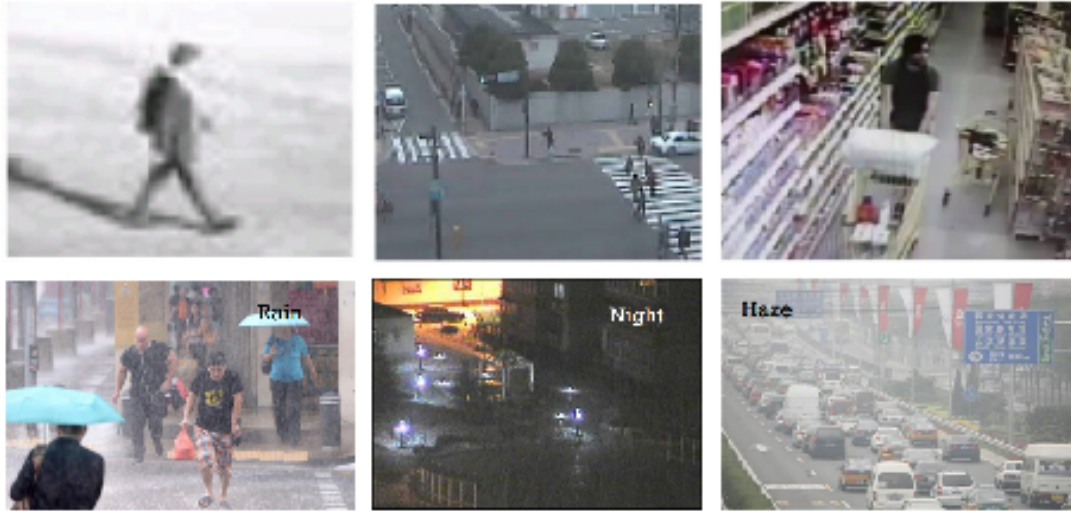
Assumption: Training data is a good representation of the testing



In the real world:

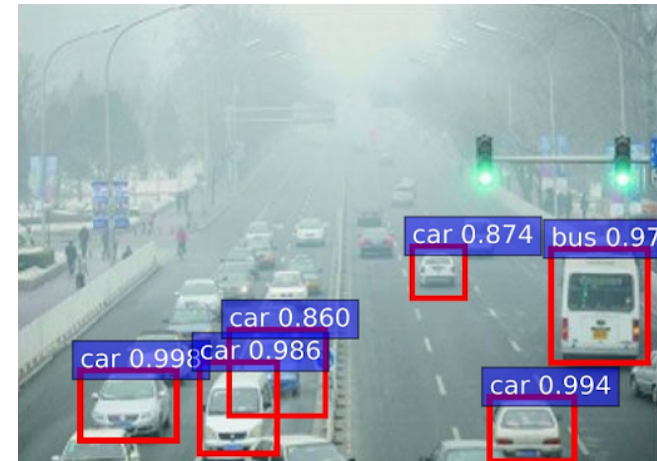


# Failure Mode I: Data Violate Assumptions



**Degraded Visual Environments (DVEs):** low-resolution, rain, low-light, haze ...

- ... cause degradations for visual understanding: reduced contrasts, detail occlusions, abnormal illumination, faded surfaces and color shift...
- It is related to, but not just, image restoration

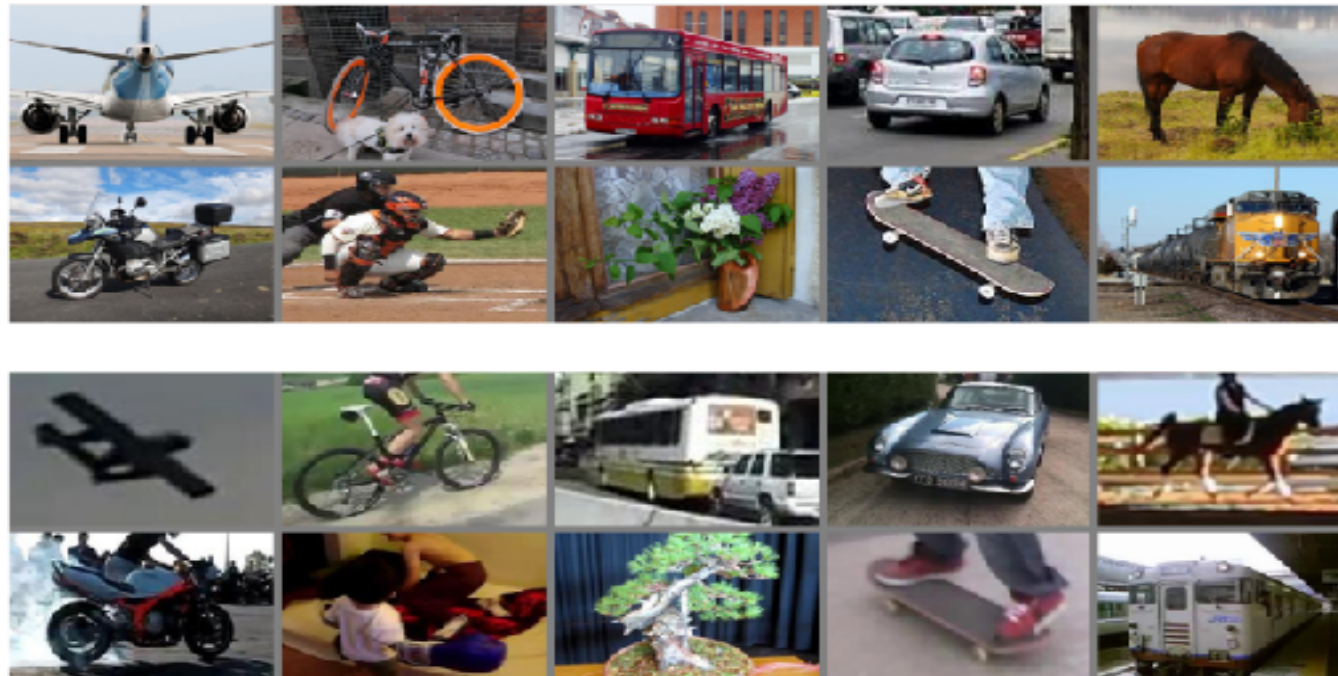


# Failure Mode I: Data Violate Assumptions

Synthetic:  
(Training)

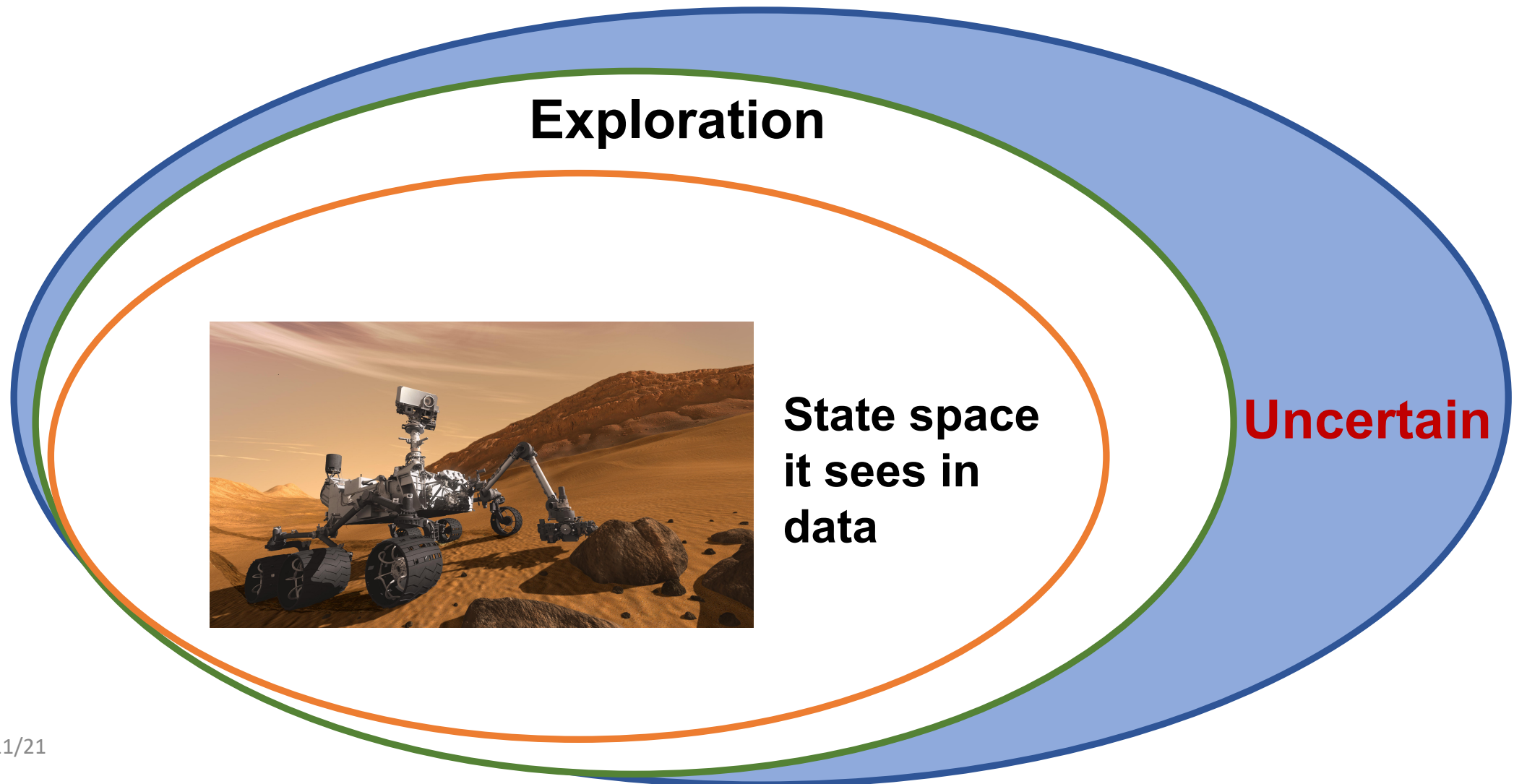


Real World:  
(Testing)

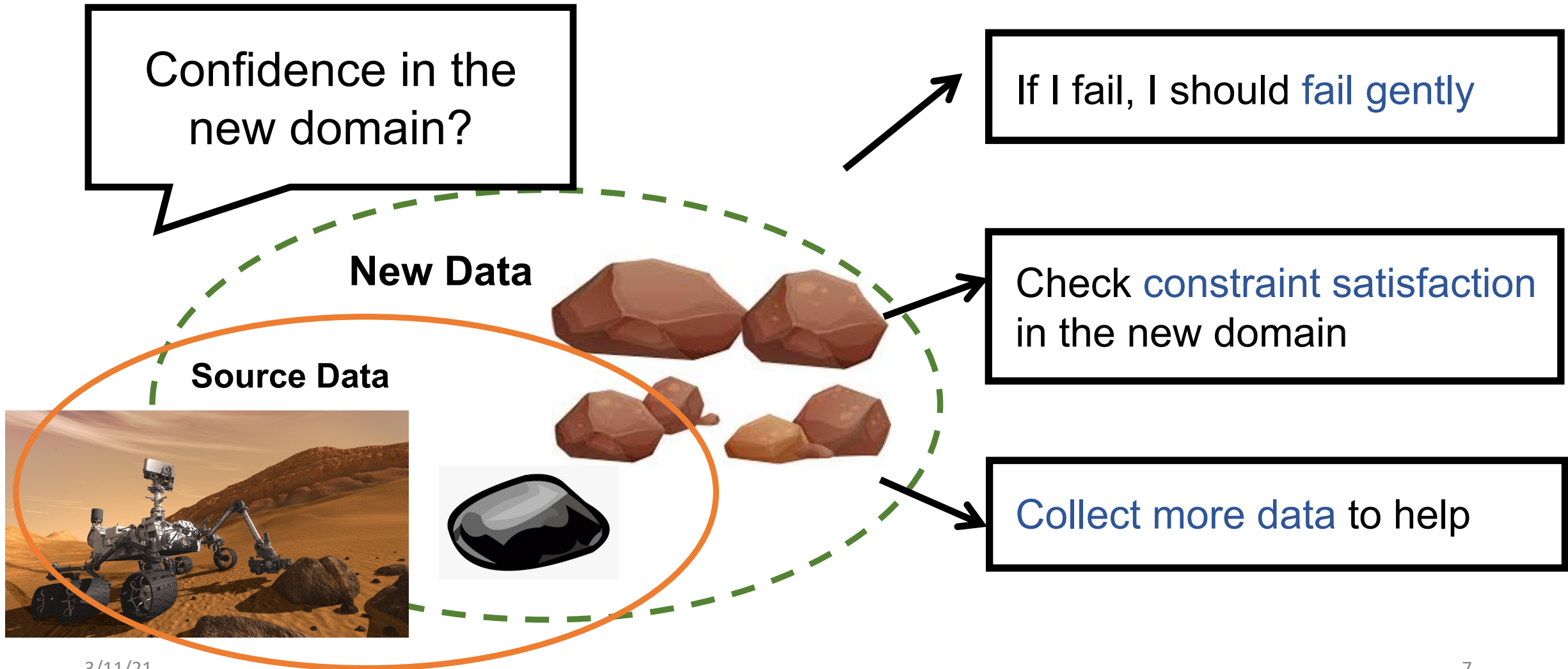


} Distribution Shift

# Failure Mode II: Exploration into Unseen Domain



# Key: Extrapolation and Model Confidence



# Failure Mode III: Malicious Adversary

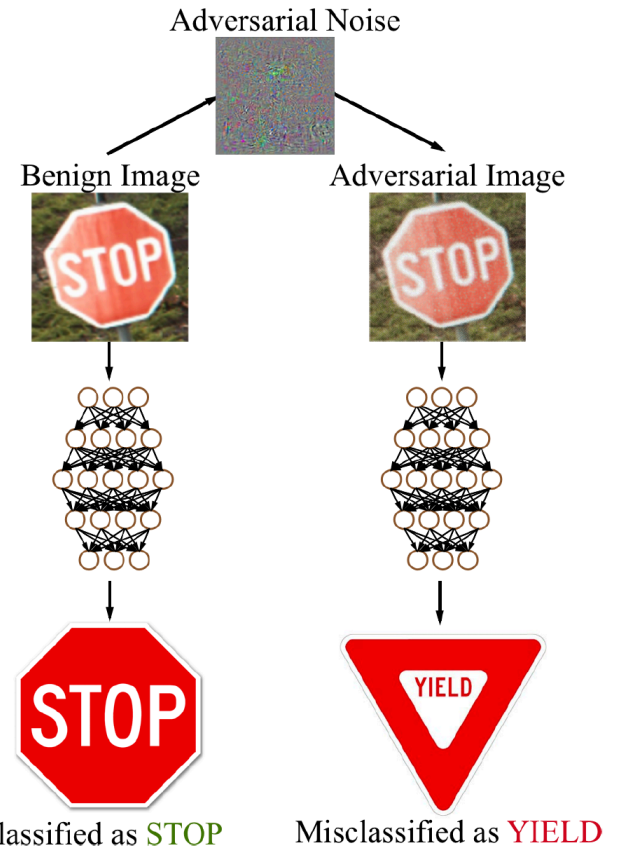
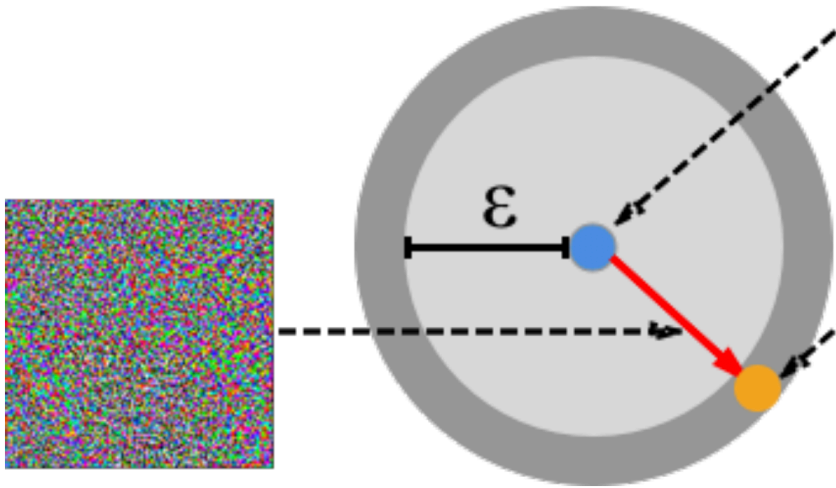
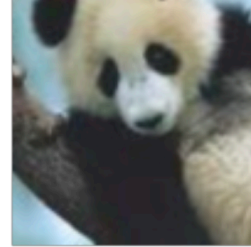
$$\underset{\|\delta\| \leq \epsilon}{\text{maximize}} \ell(h_{\theta}(x + \delta), y).$$

$$\underset{\theta}{\text{minimize}} \frac{1}{|S|} \sum_{x, y \in S} \max_{\|\delta\| \leq \epsilon} \ell(h_{\theta}(x + \delta), y).$$

57.7% panda



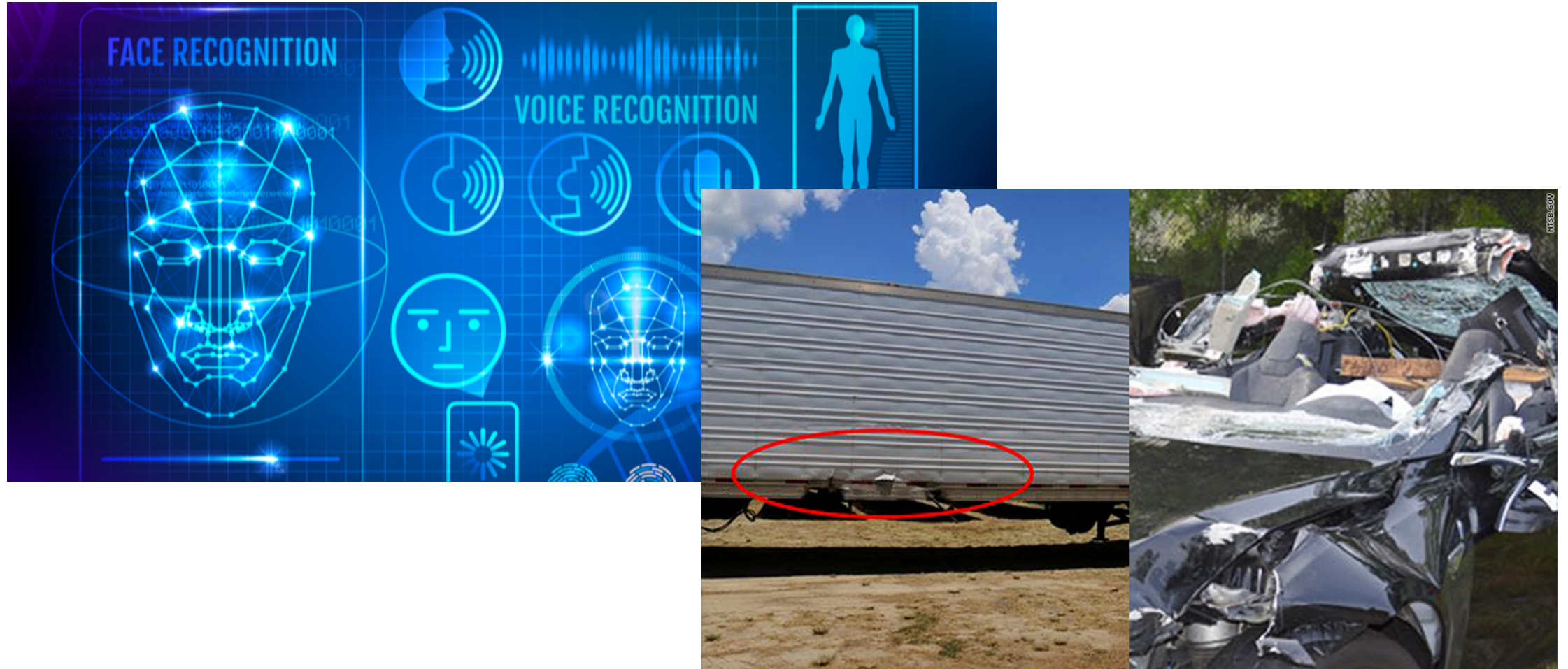
93.3% gibbon



Goodfellow et al, "Explaining and Harnessing Adversarial Examples", ICLR 2015.



# Failure Mode III: Malicious Adversary



# Research Questions:

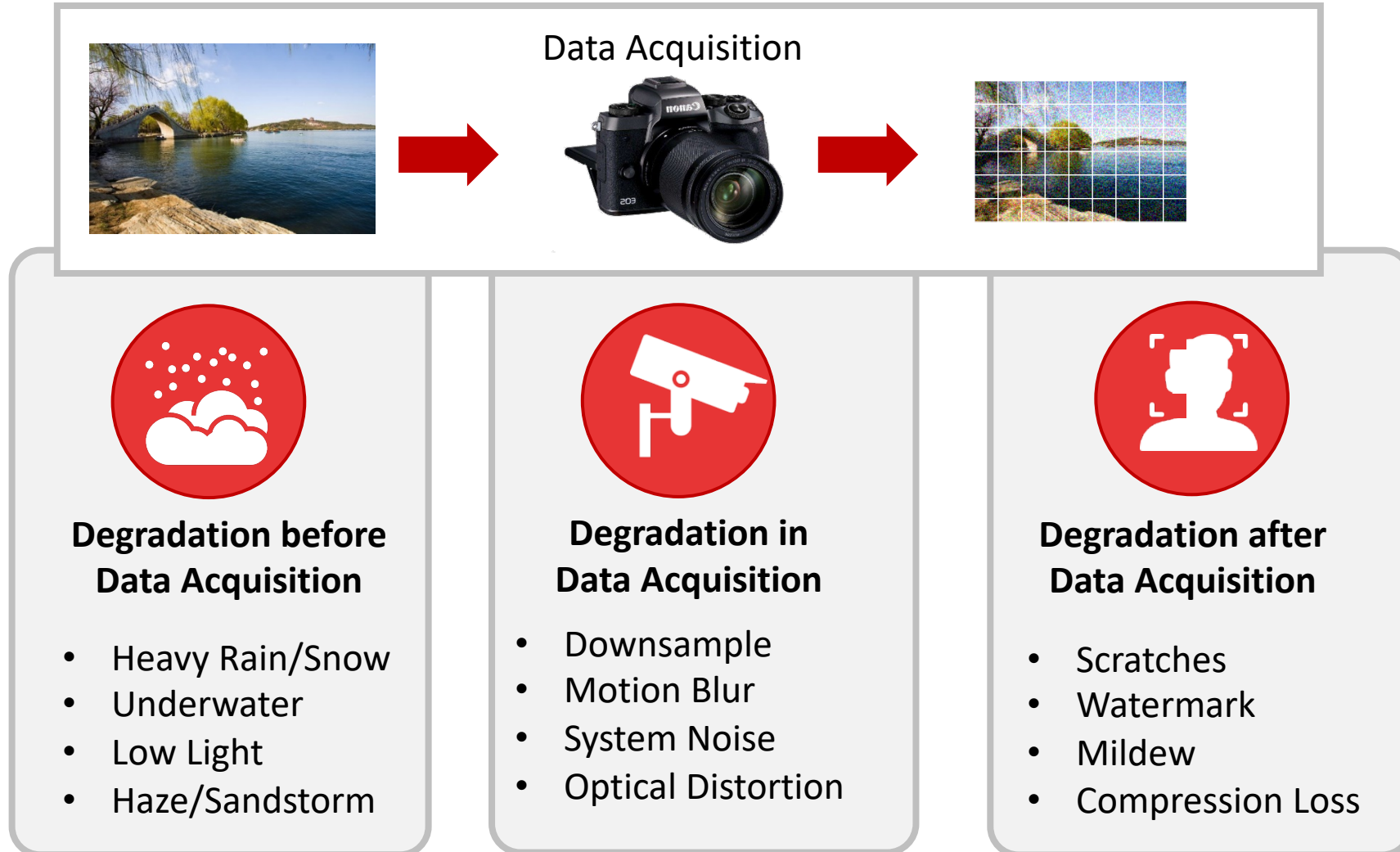
*How to produce **robust extrapolation** under various **unexpected** distribution shifts in computer vision?*

We will go through three possible answers:

- **Data-level:** preprocessing images
- **Model-level:**
  - Domain adaptation and generalization
  - Adversarial defense



# Visual Degradation



# Restoration and Enhancement: Tons of Tasks



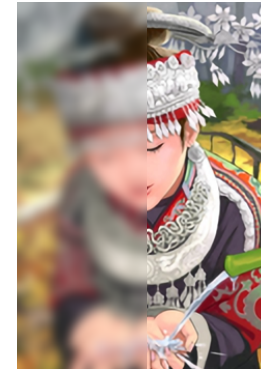
**Underwater Enhancement**



**Dehazing**



**Inpainting**



**Super Resolution**



**Rain Removal**



**Denoising**



**Low Light Enhancement**

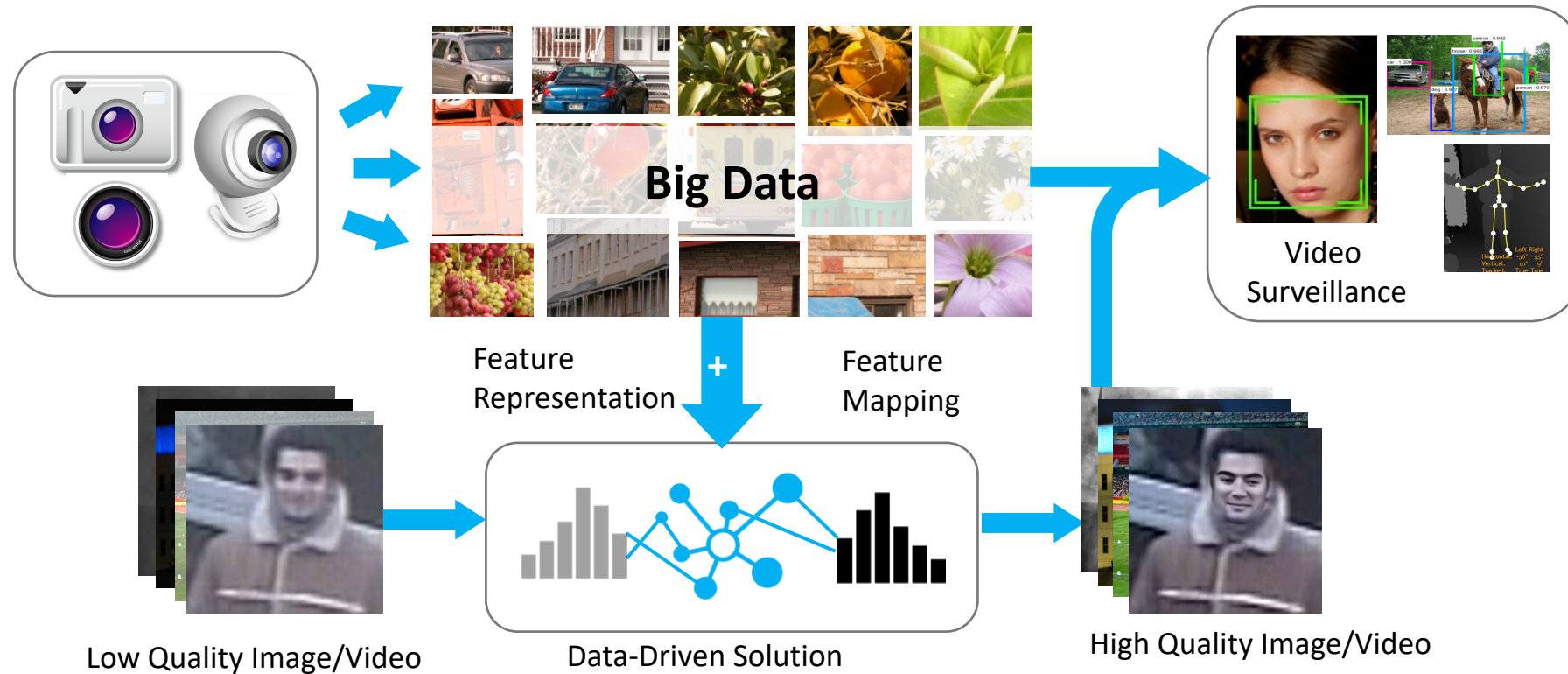


# Goal of Image Enhancement Diversified

- From traditional **signal processing (reconstruction)** viewpoint
  - Full-reference metrics: PSNR, SSIM, etc.
- ... to **human perception (subjective quality)-based**
  - No-reference metrics (e.g., NIQE), and human study
- ... And to **task-oriented, “end utility”-based**
  - Typical examples: dehazing, deraining, (extreme) light, underwater ...
  - **Representative datasets: RESIDE** dehazing (TIP'18), **MPID** deraining (CVPR'19)
  - **CVPR UG2+ Challenge:** <http://www.ug2challenge.org>

# Learning to Enhance Images

- Data-driven training of “end-to-end” models (usually assuming “pairs”)
- Prior/physical information can still be helpful



# Image Denoising

- Simplest Low-Level Vision Problem

- Noisy Measurement:

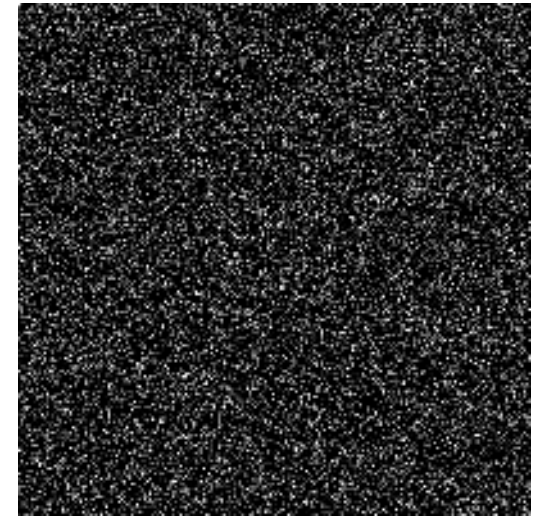
$$y = x + e$$



=



+



# Image Denoising

- Simplest Low-Level Vision Problem

- Estimate the clean image:  $\hat{x} = f(y)$





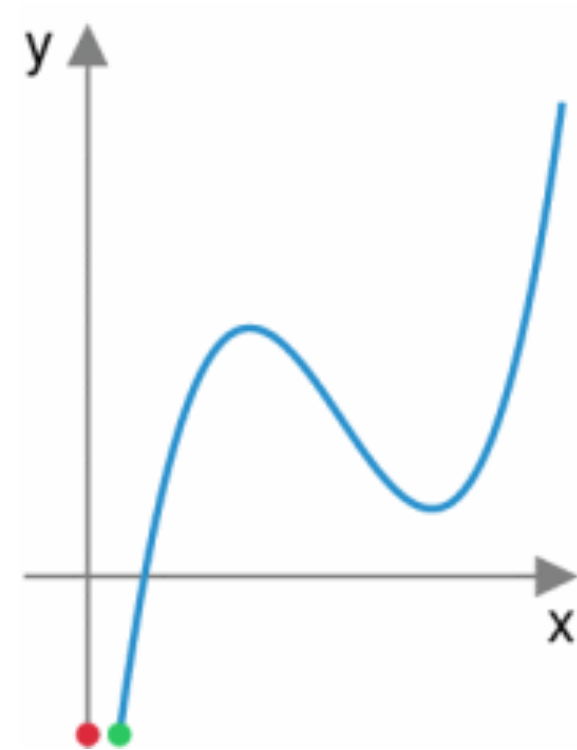
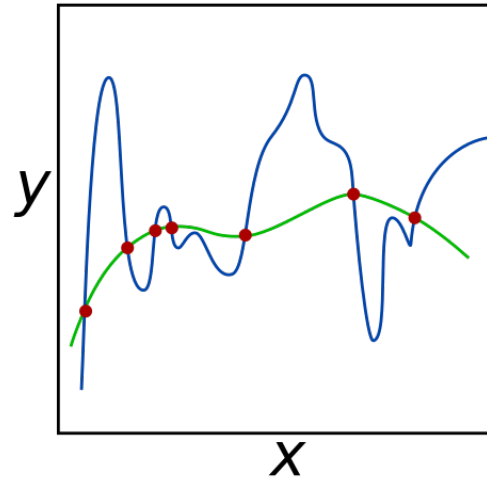
# Image Denoising – Conventional Methods

- Collaborative Filtering
  - Non-local Mean, BM3D, etc



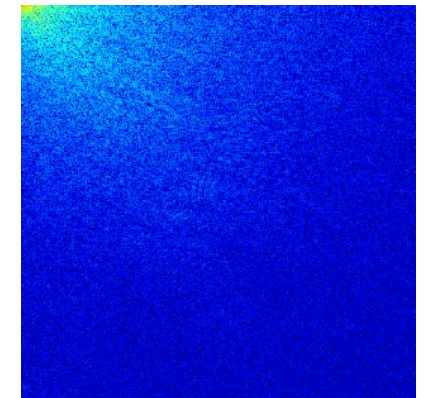
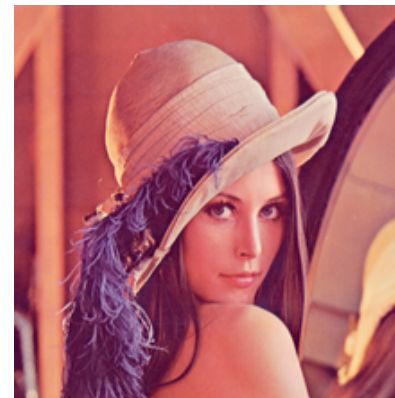
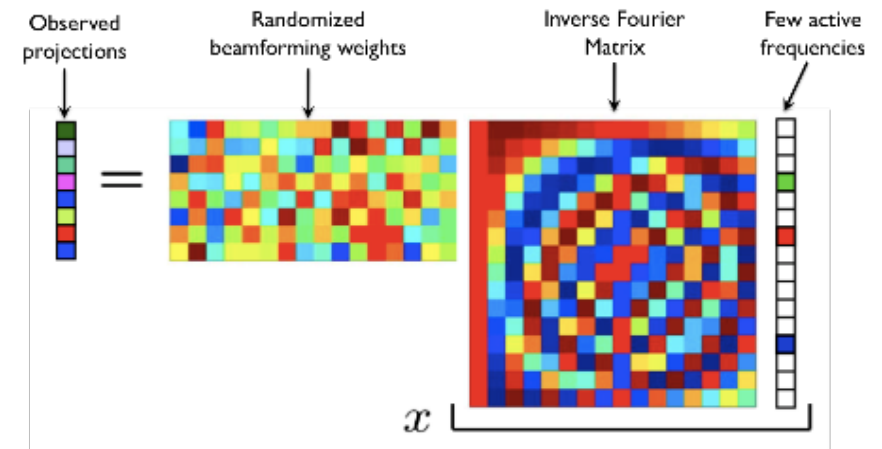
# Image Denoising – Conventional Methods

- Collaborative Filtering
  - Non-local Mean, BM3D, etc
- Piece-wise Smooth
  - Total Variation, Tikhonov Regularization, etc



# Image Denoising – Conventional Methods

- Collaborative Filtering
  - Non-local Mean, BM3D, etc
- Piece-wise Smooth
  - Total Variation, Tikhonov Regularization, etc
- Sparsity
  - Discrete Cosine Transform (DCT), Wavelets, etc
  - Dictionary Learning: KSVD, OMP, Lasso, etc
  - Analysis KSVD, Transform Learning, etc



## *Conventional*

- **Shallow Model**
  - Equivalently one free layer

## *Deep Learning*

- **Deep Model**
  - Multiple free layers



## *Conventional*

- **Shallow Model**
  - One free layer
- **Unsupervised**
  - No training corpus needed
  - Data efficient

## *Deep Learning*

- **Deep Model**
  - Multiple free layers
- **Supervised**
  - Training corpus needed
  - Data inefficient



## *Conventional*

- **Shallow Model**
  - One free layer
- **Unsupervised**
  - No training corpus needed
  - Data efficient
- **Inverse Problem**
  - Assumption & Understanding of the Data
  - Regularizer & structures of the Model
  - Flexible

## *Deep Learning*

- **Deep Model**
  - Multiple free layers
- **Supervised**
  - Training corpus needed
  - Data inefficient
- **Inverse Problem**
  - Little assumption
  - Almost free model
  - Few work until recent



# Image Denoising by Deep Learning

- **Natural Idea:** train a denoising autoencoder, that regresses clean images from noisy ones
- It is not easy for deep networks to outperform classical methods such as BM3D!!
  - BM3D is shown to be better at dealing with self-repeating regular structures
- **How to outperform BM3D using a deep network denoiser? Some verified tips:**
  - The model richness is large enough, i.e. enough hidden layers with sufficiently many hidden units.
  - The patch size is chosen large enough, i.e. a patch contains enough information to fit a complicated denoising function that covers the long tail.
  - The chosen training set is large enough
- **Other benefits of deep network denoiser:**
  - The testing speed of deep networks is much faster than BM3D, KSVD etc., benefiting from GPU.
  - Deep networks can be generalized to other noise types, if correctly supplied in training.
- Recent works show great progress!
  - **Check out Git repo:** <https://github.com/wenbihan/reproducible-image-denoising-state-of-the-art>

# Image Denoising by Deep Learning

- Reference: *“Image denoising: Can plain Neural Networks compete with BM3D?”*



clean (name: 008934)



noisy ( $\sigma = 25$ )PSNR:20.16dB



BM3D: PSNR:29.65dB



ours: PSNR:**30.03**dB



clean (name: barbara)



noisy ( $\sigma = 25$ )PSNR:20.19dB



BM3D: PSNR:**30.67**dB



ours: PSNR:29.21dB



# Image Deblurring

- Blurred Measurement:

$$y = M \otimes x$$



=



$\otimes$

$$M = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

# Image Deblurring

- Estimate the stable image:  $\hat{x} = f(y)$



# Image Deblurring

- Non-blind Image Deblurring
  - Suppose you know the blurring kernel,  $M$ .
  - $\hat{x} = f(y, M)$
  - All training data need to have consistent  $M$ , as the testing data

# Image Deblurring

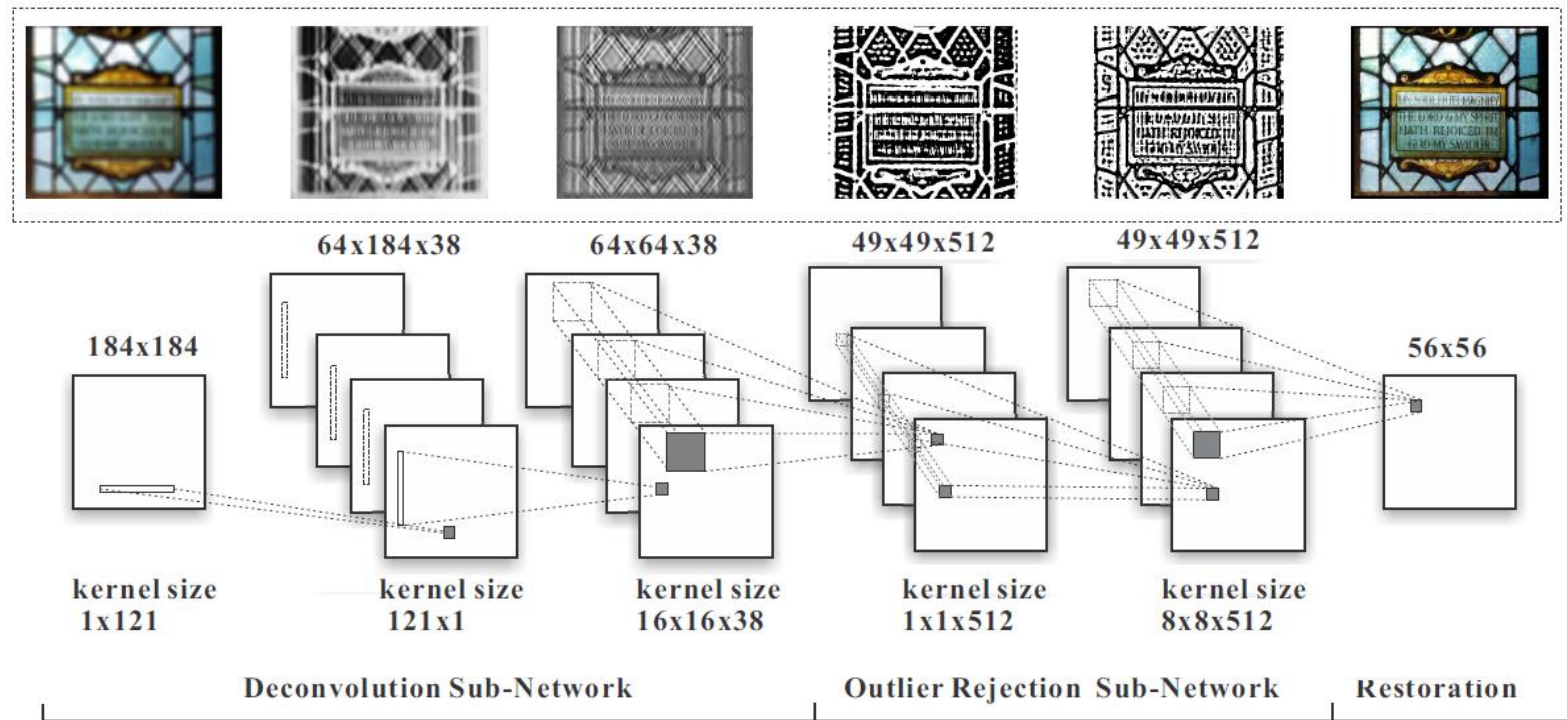
- Non-blind Image Deblurring
  - Suppose you know the blurring kernel,  $M$ .
  - $\hat{x} = f(y, M)$
  - All training data need to have consistent  $M$ , as the testing data
- Blind Image Deblurring – More challenging yet practical problem
  - Estimate both the image, and the blurring kernel
  - $\{\hat{x}, M\} = f(y)$

# Image Deblurring by Deep Learning

Reference: “*Deep convolutional neural network for image deconvolution*”

- **Key Technical Features:**

- Treat deblurring as a deconvolution task, and the deconvolution operation can be approximated by a convolutional network with very large filter sizes
- Concatenation of deconvolution CNN module with another denoising CNN module to suppress artifacts and reject outliers

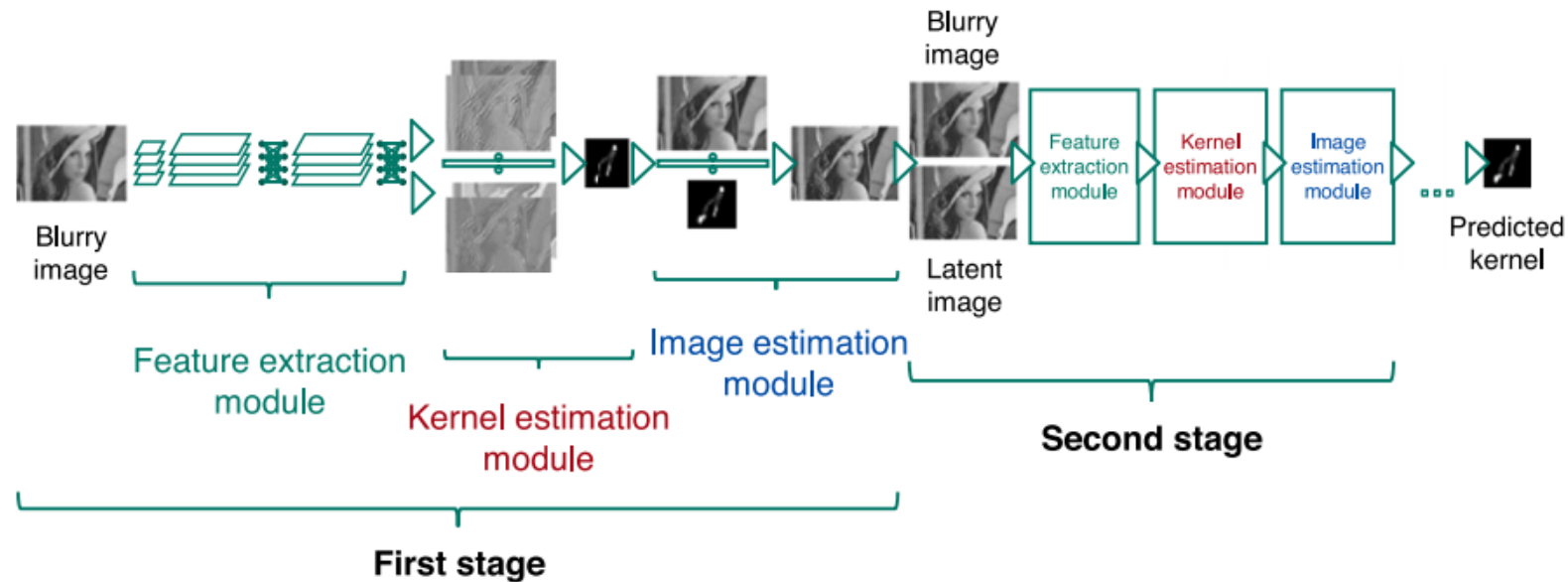


# Blind Image Deblurring by Deep Learning

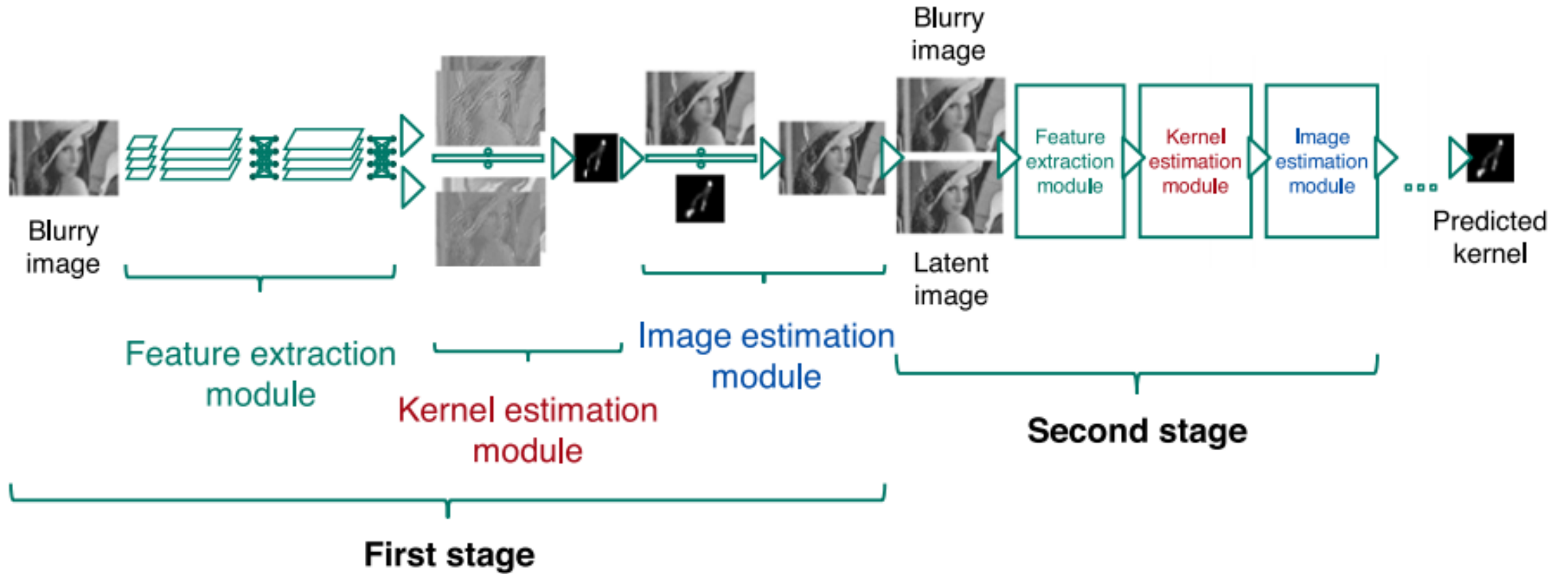
Reference: *“Learning to Deblur”*

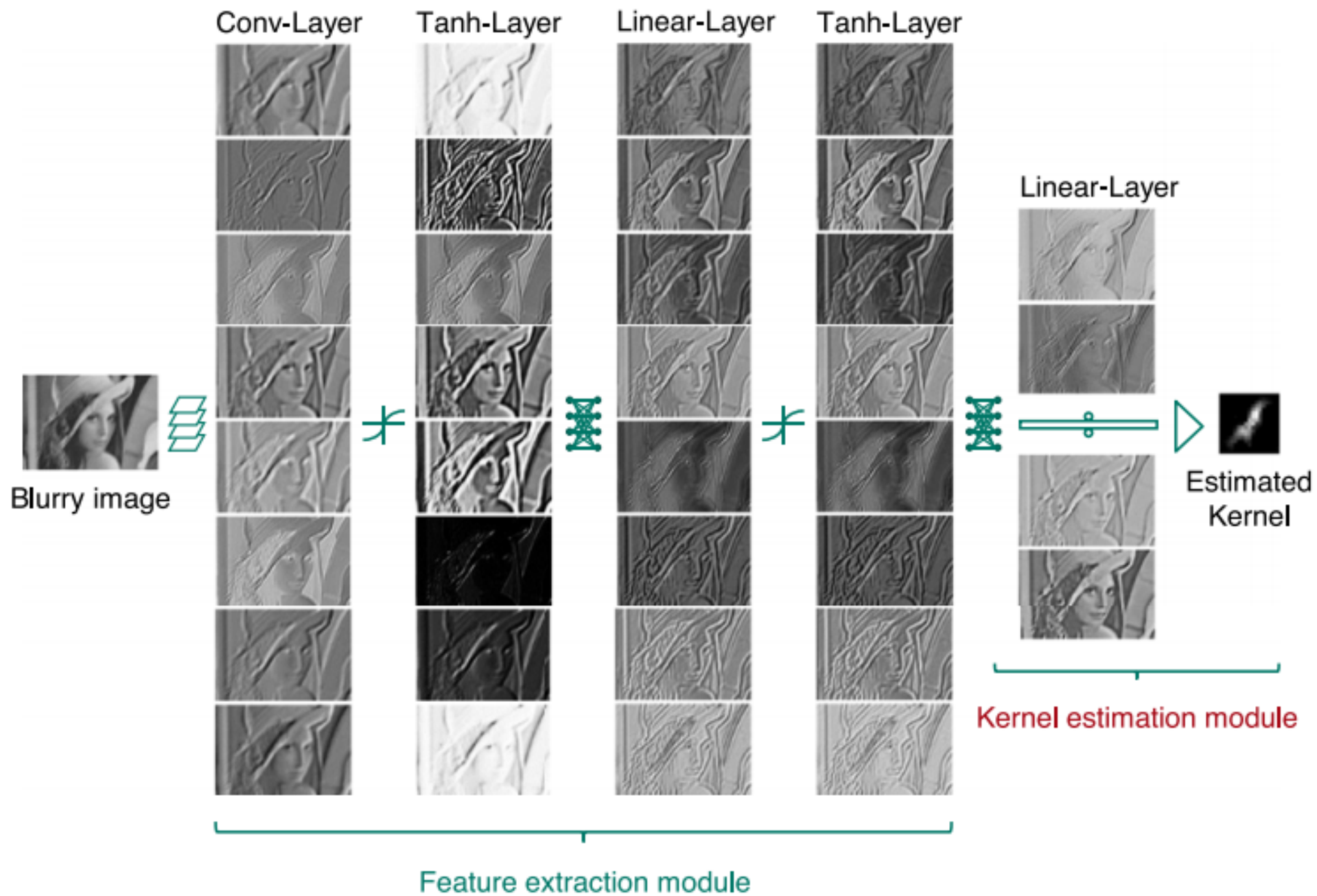
- **Key Technical Features:**

- Iteratively estimate both the blurring kernel, and the underlying image.
- Separate network to capture the image property, and the kernel information, respectively.



# Blind Image Deblurring







# Blind Image Deblurring



PSNR 24.13 dB



PSNR 24.61 dB



PSNR 24.96 dB



PSNR 20.71 dB

Deblurring result of  
Xu et al. [20]



PSNR 23.87 dB

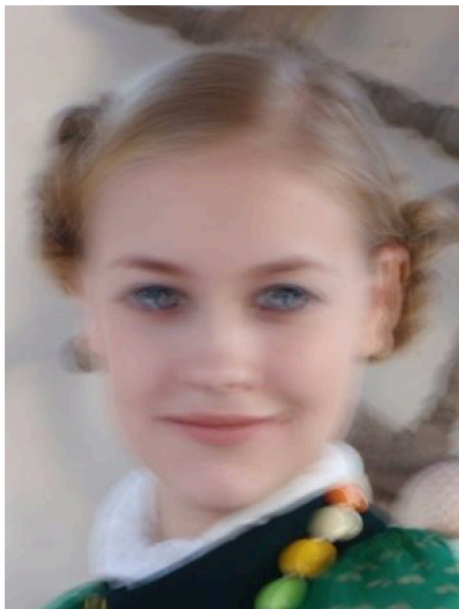
Deblurring result with  
content agnostic training



PSNR 26.32 dB

Deblurring result with  
content specific training

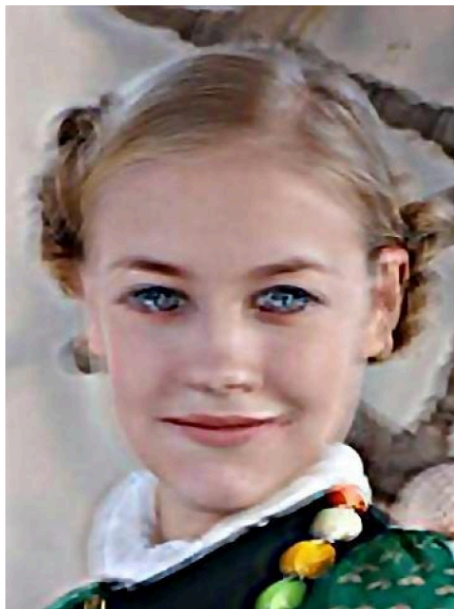
Blurry image with  
ground truth kernel



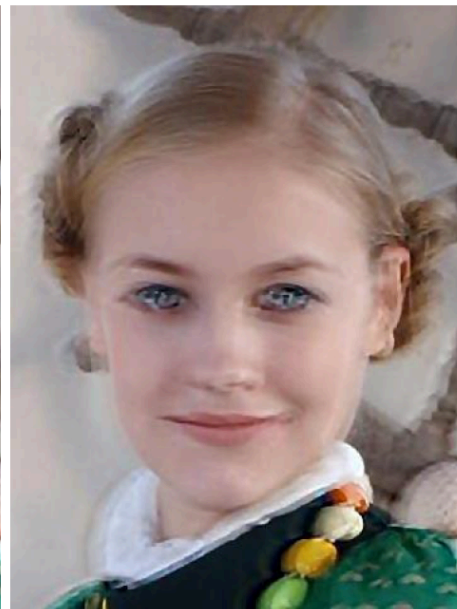
(a) Blurred photo



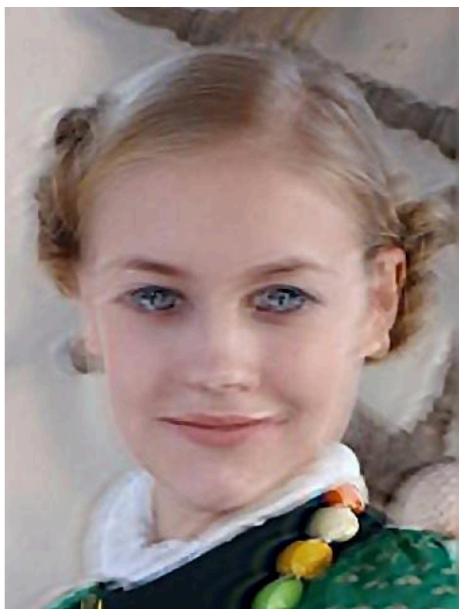
(b) Whyte *et al.* [40]



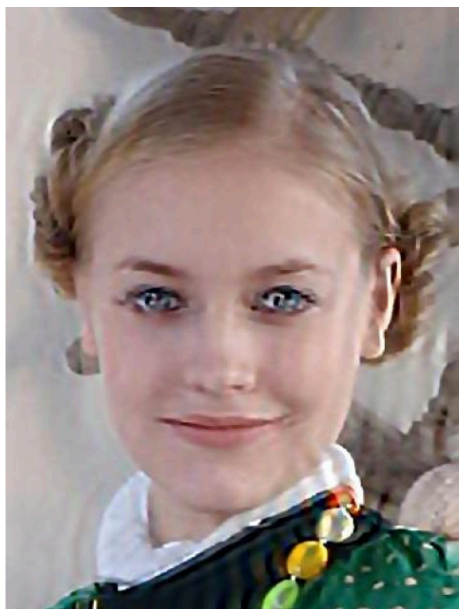
(c) Krishnan *et al.* [14]



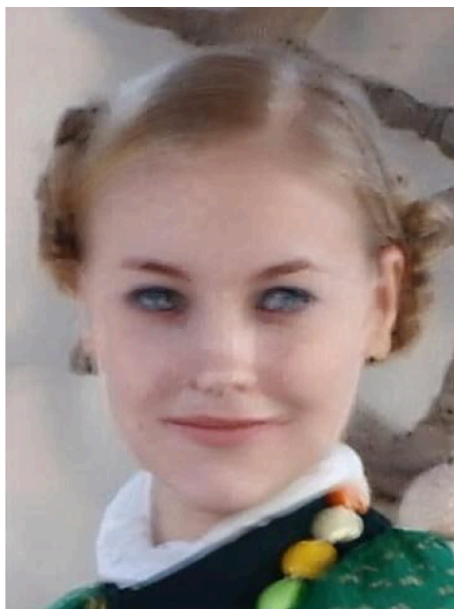
(d) Sun *et al.* [18]



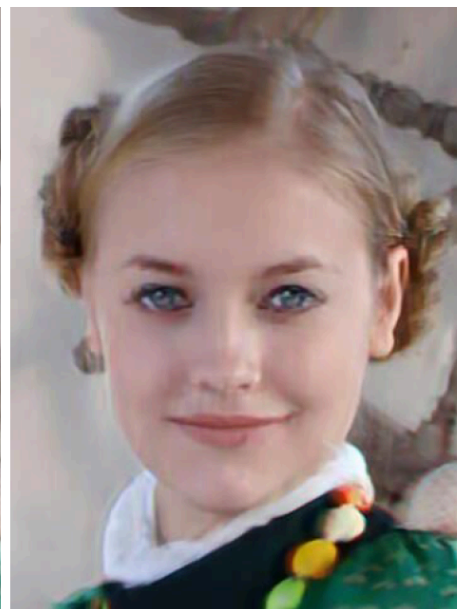
(e) Xu *et al.* [33]



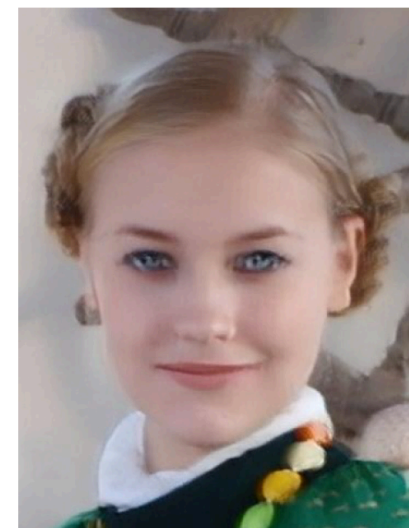
(f) Pan *et al.* [15]



(g) DeepDeblur [2]



(h) SRN [3]



DeblurGAN V2 (2019)

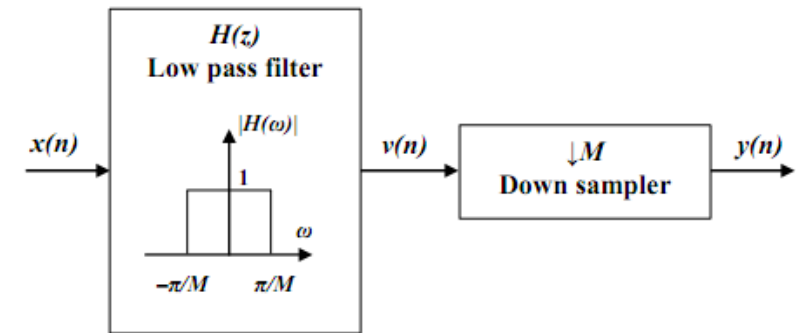
# Image Super-Resolution

- Low-Resolution Measurement:

$$y = D * M \otimes x$$



=



# Image Super-Resolution

- Estimate the stable image:  $\hat{x} = f(y)$

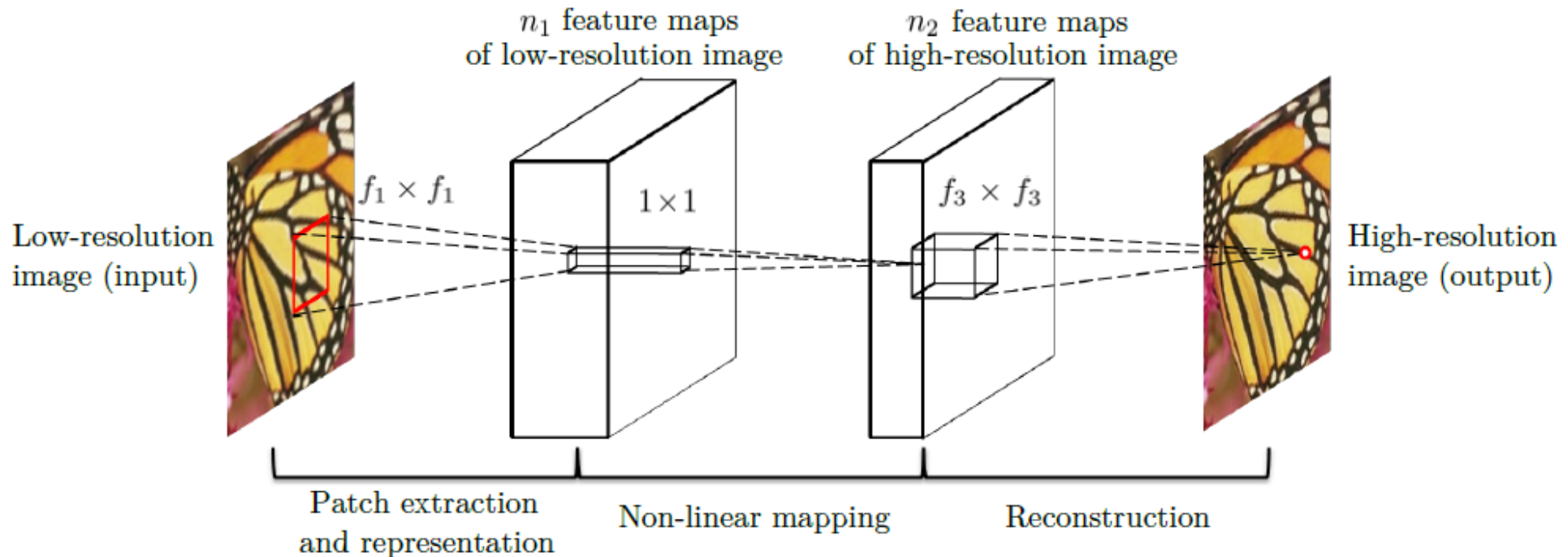


# Image Super Resolution by Deep Learning

Reference: "Image super-resolution using deep convolutional networks"

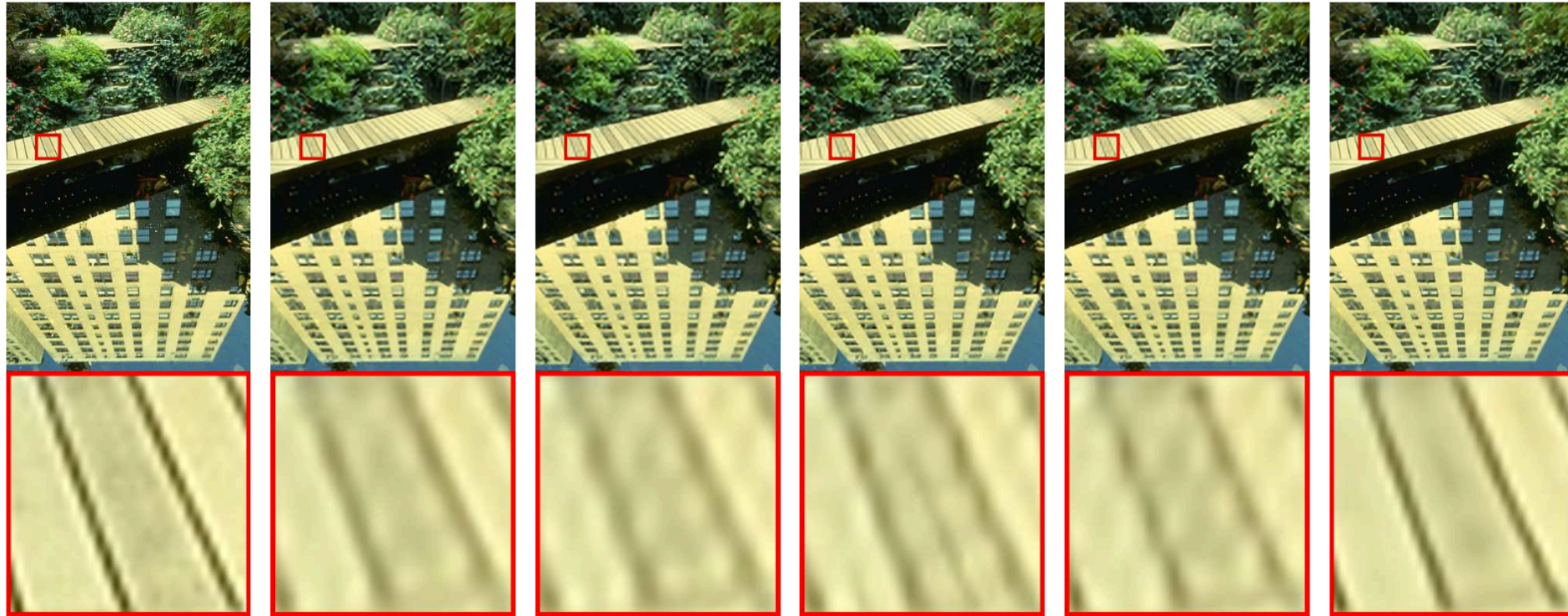
- Key Technical Features:

- Learns an end-to-end mapping from low to high-resolution images as a deep CNN
- Closely mimic the traditional SR pipeline: LR feature extraction -> coupled LR-HR feature space mapping -> HR image reconstruction



# Image Super Resolution by Deep Learning (2013 – 2017)

Super-resolution results of “148026” (*B100*) with scale factor  $\times 3$  (from VDSR paper)



Ground Truth  
(PSNR, SSIM)

A+ [22]  
(22.92, 0.7379)

RFL [18]  
(22.90, 0.7332)

SelfEx [11]  
(23.00, 0.7439)

SRCNN [5]  
(23.15, 0.7487)

VDSR (Ours)  
(23.50, 0.7777)

# New Trends?

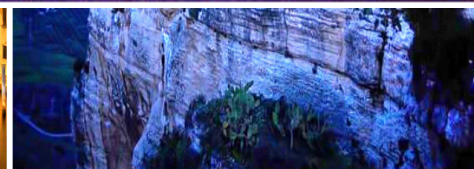
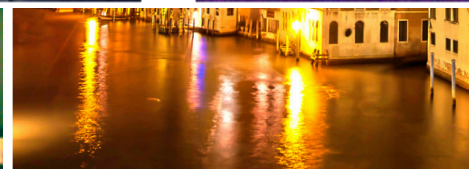
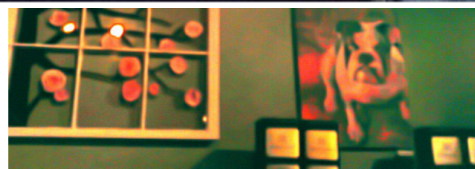
- **New topic:** dehazing, deraining, low light enhancement, etc.
- **New goal:** human perception **v.s.** machine consumption
- **New setting:** from supervised to unsupervised training (no “GT”)
  - ... or relying on “synthetic pairs”
- **New domain:** medical images, infrared images, remote sensing images, etc.
- **New concern:** “All-in-one” adaptivity, efficient implementation, etc.

# Lots of Progress – but “not there” yet

**Example:**



**Why?**





# Shortage of Real-World Generalization

- Most SOTA algorithms are trained with {clean, corrupted} paired data
  - Such paired training data is usually collected by synthesis (assuming known degradation model), which typically **oversimplifies** the real-world degradations
  - As a result, the trained model “overfits” simpler degradation process and generalizes poorly to real visual degradations
- Real-world collection of paired data?
  - Can be done in small scale and/or in controlled lab environments
    - e.g. some recent datasets in light enhancement, and raindrop removal
  - Very difficult to “scale up”, sometimes maybe impossible

# EnlightenGAN: Deep Light Enhancement without Paired Supervision

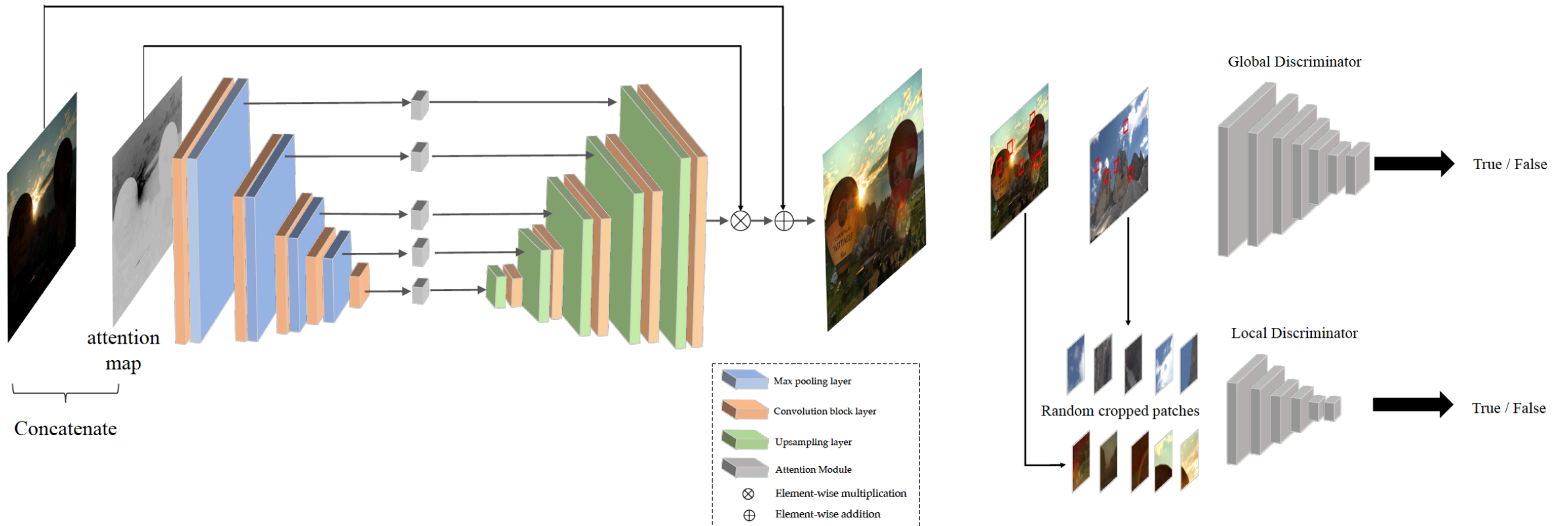


**Goal:** Light enhancement made automatic, adaptive, and artifact-free

# From Supervised to Unsupervised Enhancement

- EnlightenGAN is the first work that successfully introduces unpaired training to low-light image enhancement.
  - It only needs **one low-light set A** and **another normal-light set B** to train, while **A and B could consist of completely different images!**
- What makes **Unpaired Training** unique and attractive?
  - It removes the dependency on paired training data
    - Hence enabling us to train with massive images from different domains
  - It also avoids overfitting any specific data generation/imaging protocol
    - ...that previous works implicitly rely on, leading to stronger generalization.
  - It makes EnlightenGAN particularly easy and flexible to be adapted
    - when enhancing real-world low-light images from completely different/unseen domains

# Model Architecture



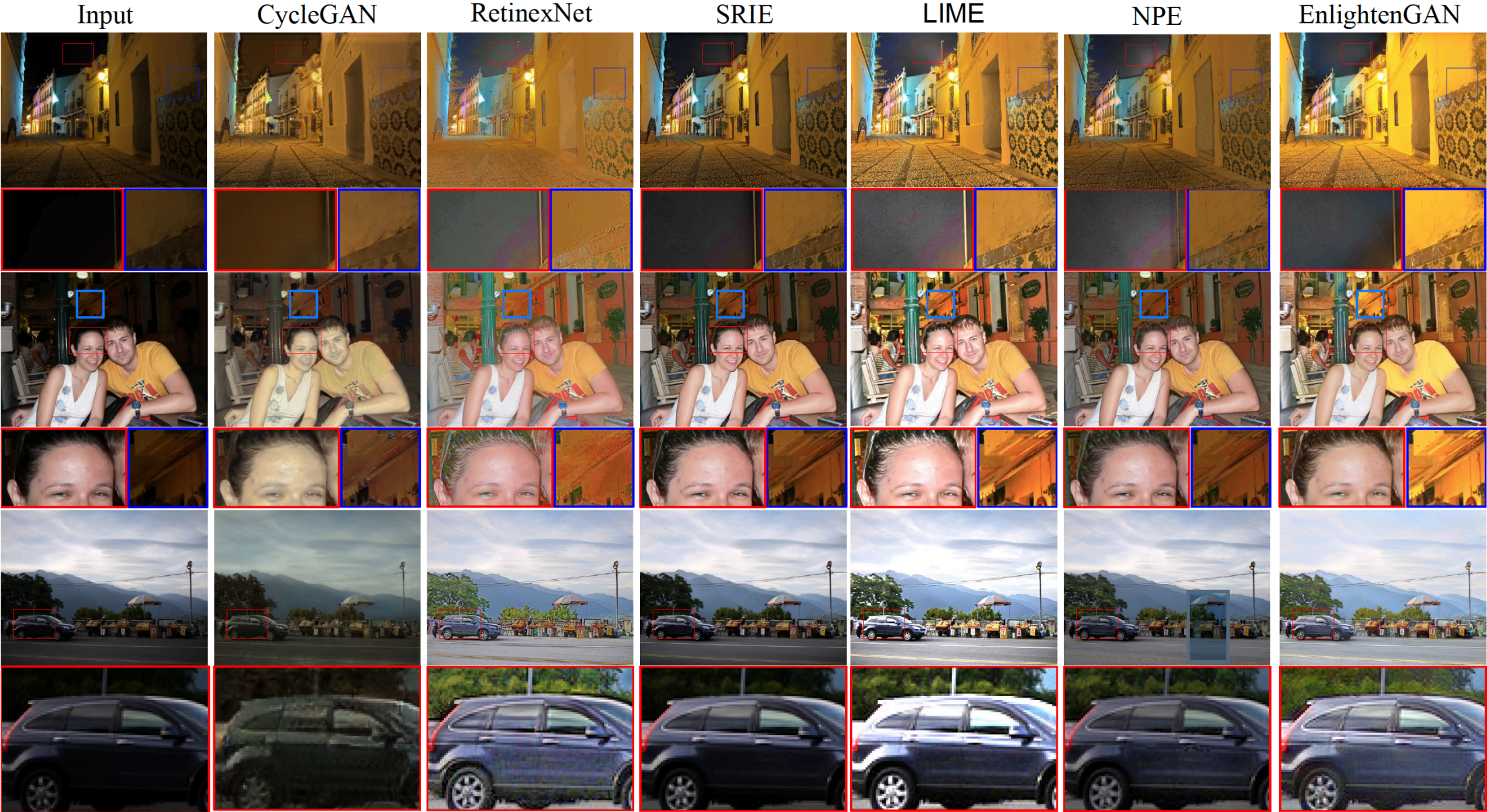
Paper: <https://arxiv.org/abs/1906.06972>

Code: <https://github.com/VITA-Group/EnlightenGAN>

# Key Components that make it work

- Using a large, real unpaired training dataset
  - We assemble a mixture of 914 low light and 1016 normal light images (no need for any pair)
  - From several datasets and HDR sources, with a wide range of image quality factors)
- Combining a global and a local patch discriminator
  - Taking care of both global composition, and local fine details
- Self Feature-Preserving Loss
  - Computing VGG distance between input-output images
  - Based on our empirical observation that VGG features are robust to light changes
- Self-Regularized Attention
  - We take the illumination channel  $I$  of the input RGB image, normalize it to  $[0,1]$ , and then use  $1 - I$  as our self-regularized attention map.
  - We then resize the attention map and multiply it with all intermediate feature maps the output.

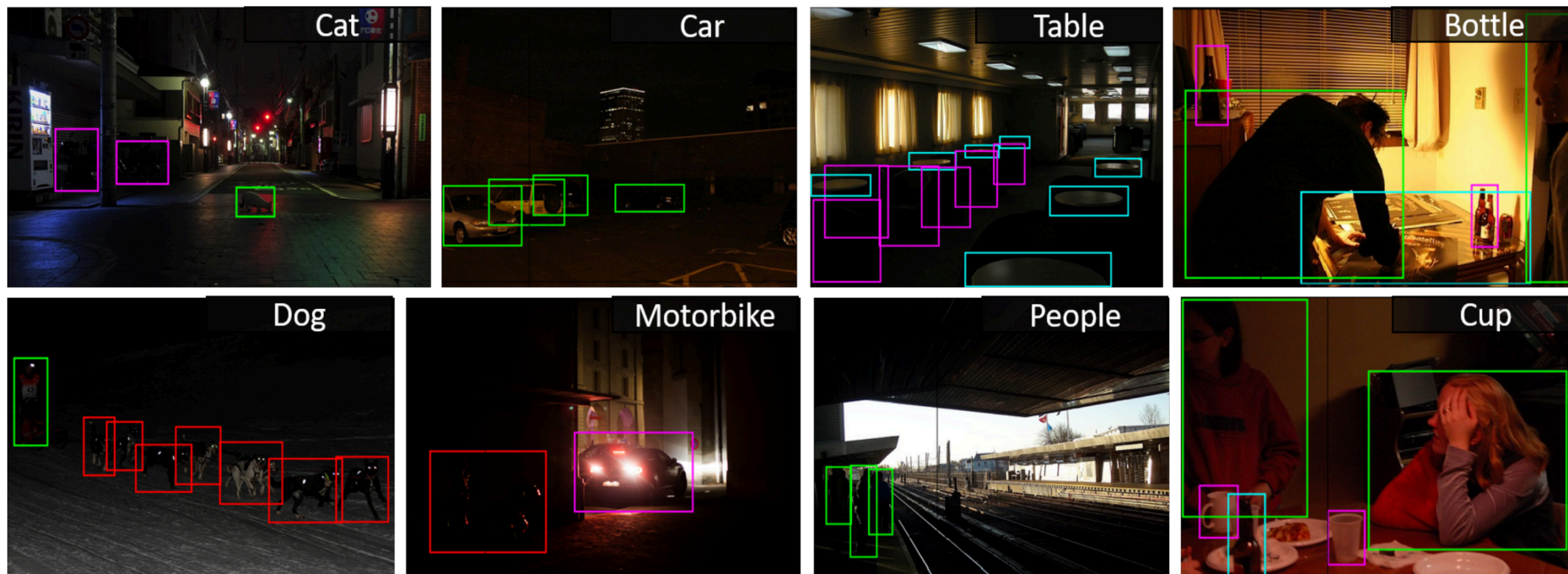
# Comparison with State-of-the-Arts



# [New!] Frustratingly Easy Adaptation to New Data



# PreProcessing for Improving Classification



- We applying our pretrained EnlightenGAN as a pre-processing step on the testing set of the ExDark dataset , followed by passing through another ImageNet-pretrained ResNet-50 classifier.
- It improves the classification accuracy from 22.02% (top-1) and 39.46% (top-5), to 23.94% (top-1) and 40.92% (top-5) after enhancement.





Uncertainty & Robustness for Out-of-Distribution Generalization

# What do we mean by Uncertainty?

Return a distribution over predictions rather than a single prediction.

- **Classification**: Output label along with its confidence.
- **Regression**: Output mean along with its variance.

Good uncertainty estimates quantify **when we can trust the model's predictions**.

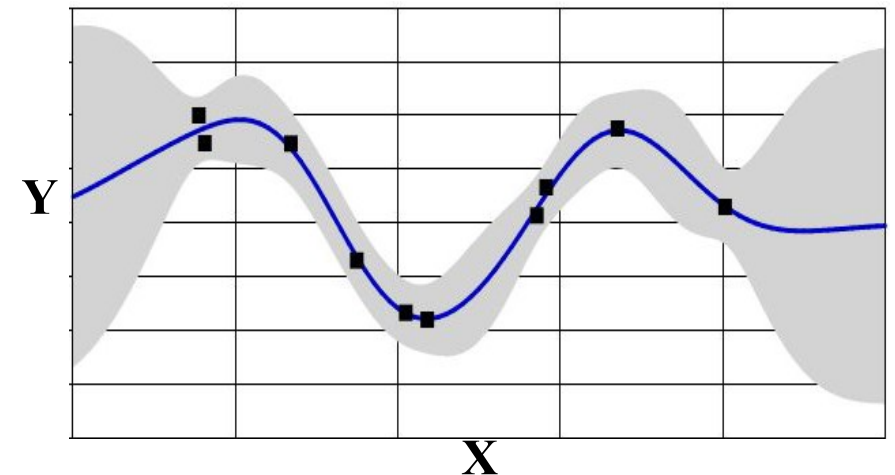
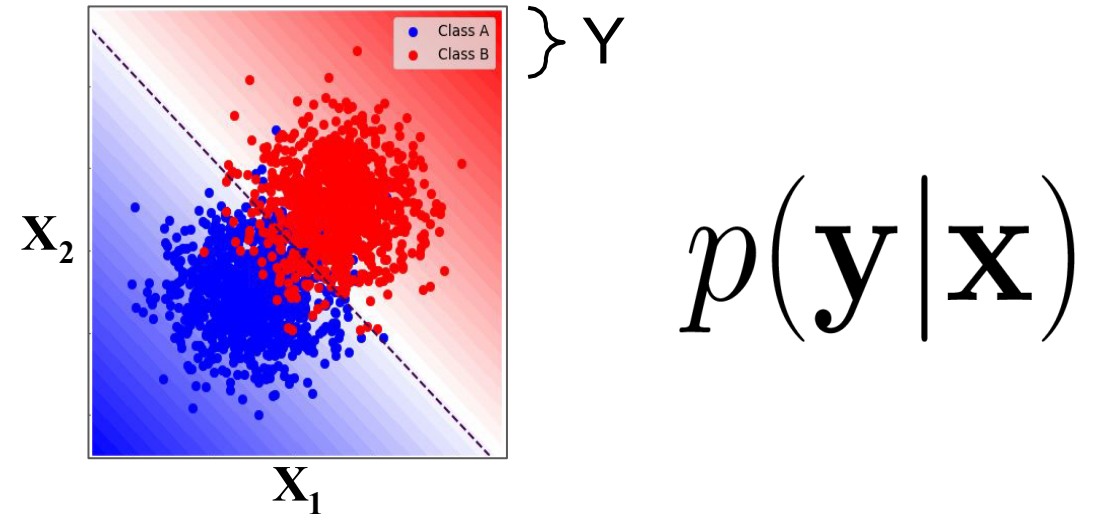


Image credit: Eric Nalisnick

# What do we mean by Out-of-Distribution Robustness?

**I.I.D.**  $p_{\text{TEST}}(y,x) = p_{\text{TRAIN}}(y,x)$

**O.O.D.**  $p_{\text{TEST}}(y,x) \neq p_{\text{TRAIN}}(y,x)$

Examples of dataset shift:

- **Covariate shift.** Distribution of features  $p(x)$  changes and  $p(y|x)$  is fixed.
- **Open-set recognition.** New classes may appear at test time.
- **Label shift.** Distribution of labels  $p(y)$  changes and  $p(x|y)$  is fixed.

# ImageNet-C: Varying Intensity for Dataset Shift

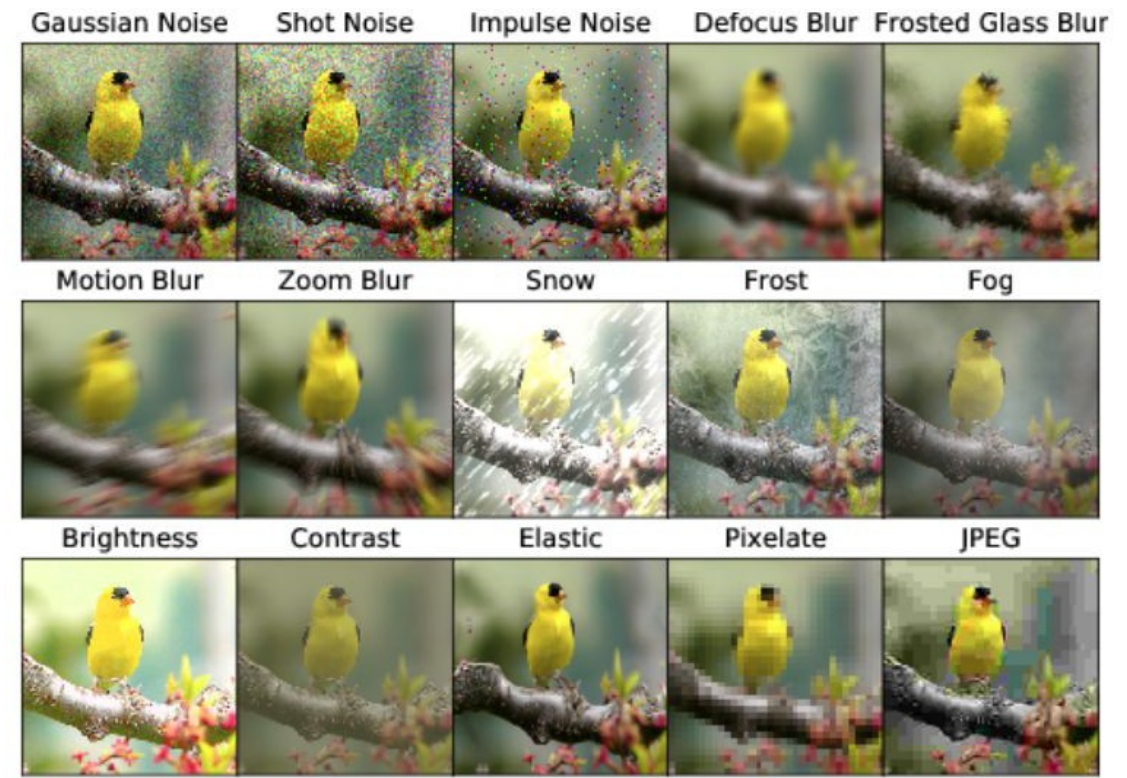
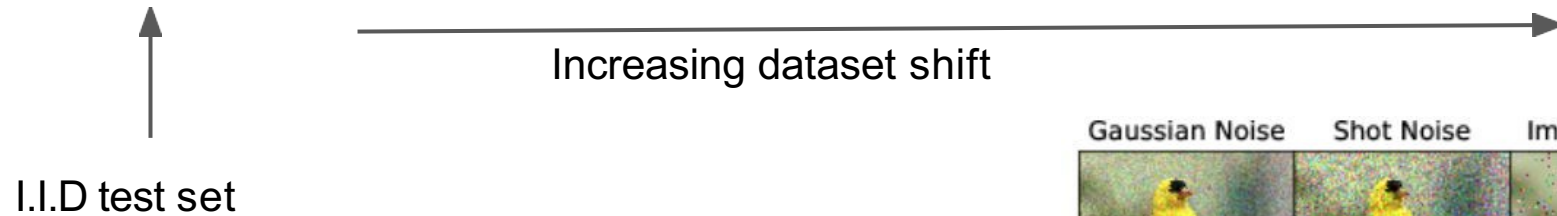
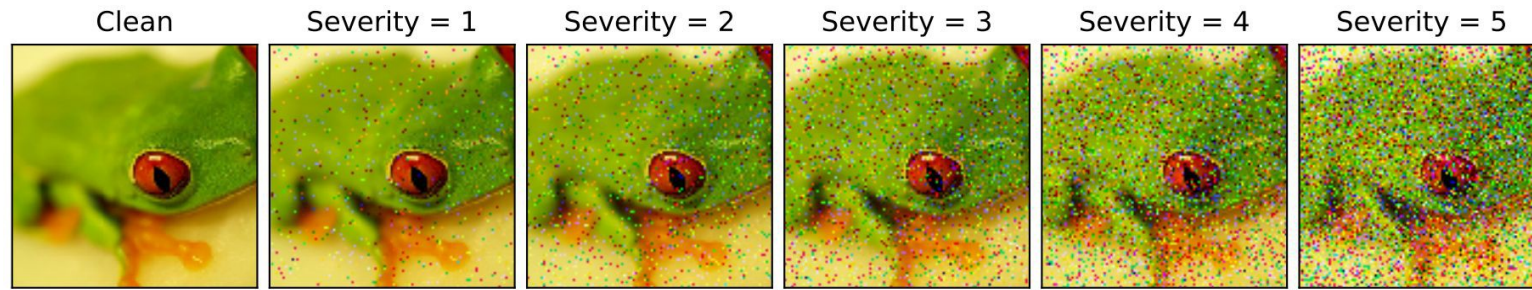
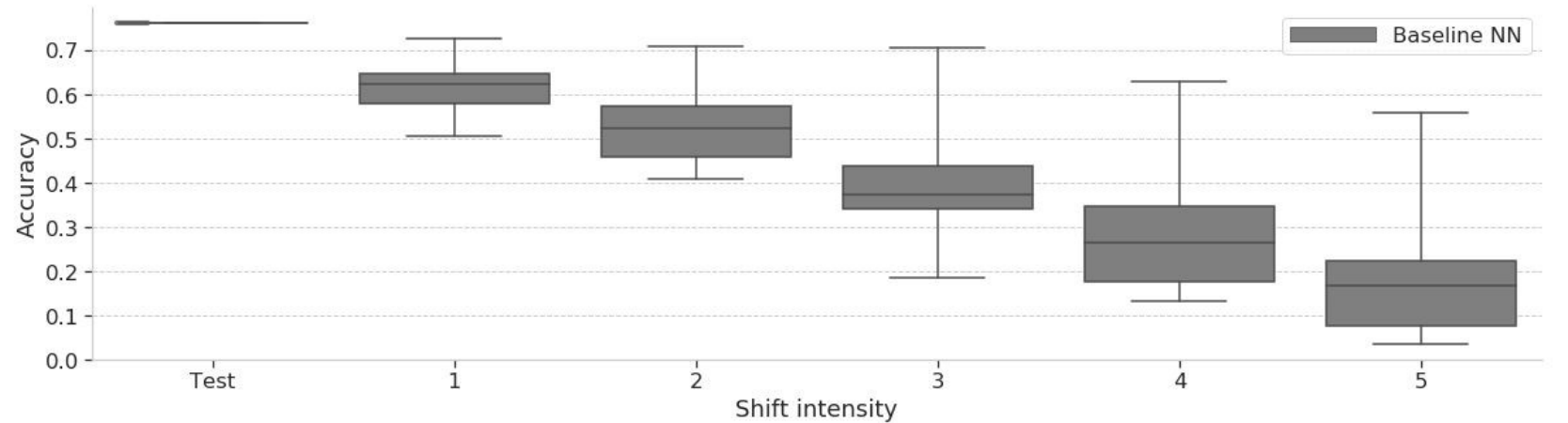
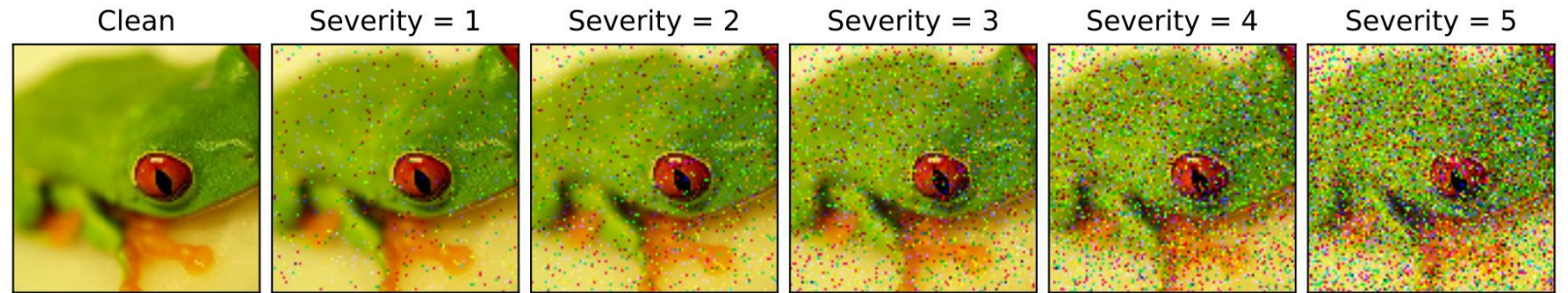


Image source: Benchmarking Neural Network Robustness to Common Corruptions and Perturbations, [Hendrycks & Dietterich, 2019](#).

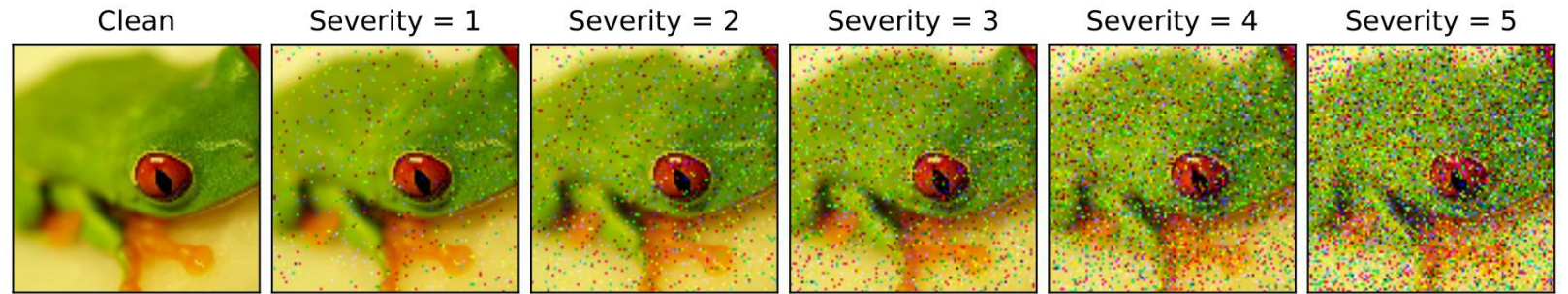
# Neural networks do not generalize under covariate shift



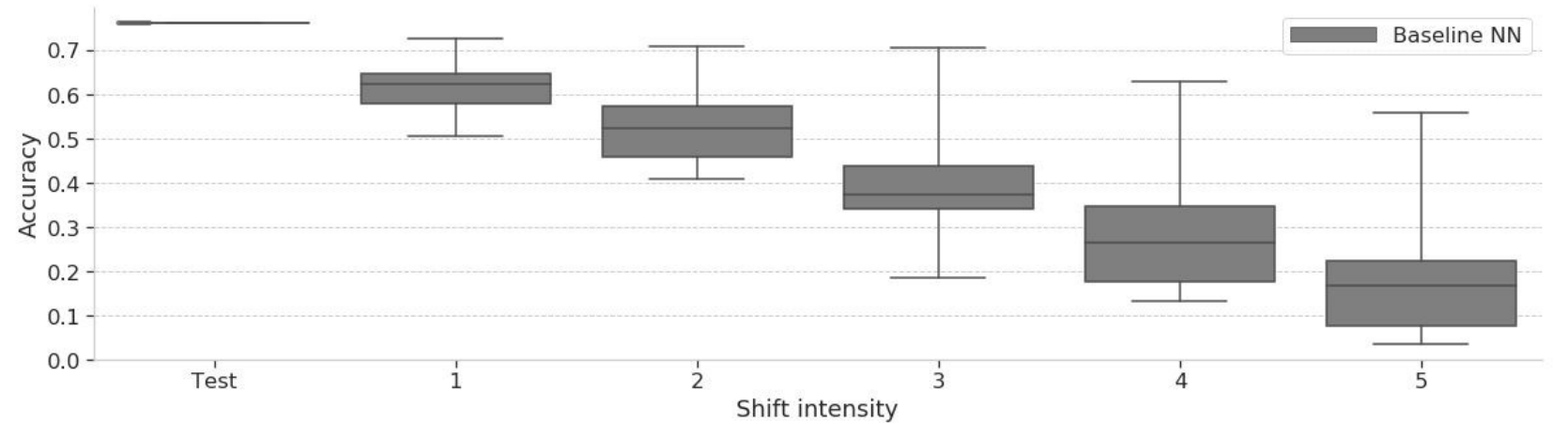
- **Accuracy drops** with increasing shift on Imagenet-C

- But do the models know that they are less accurate?

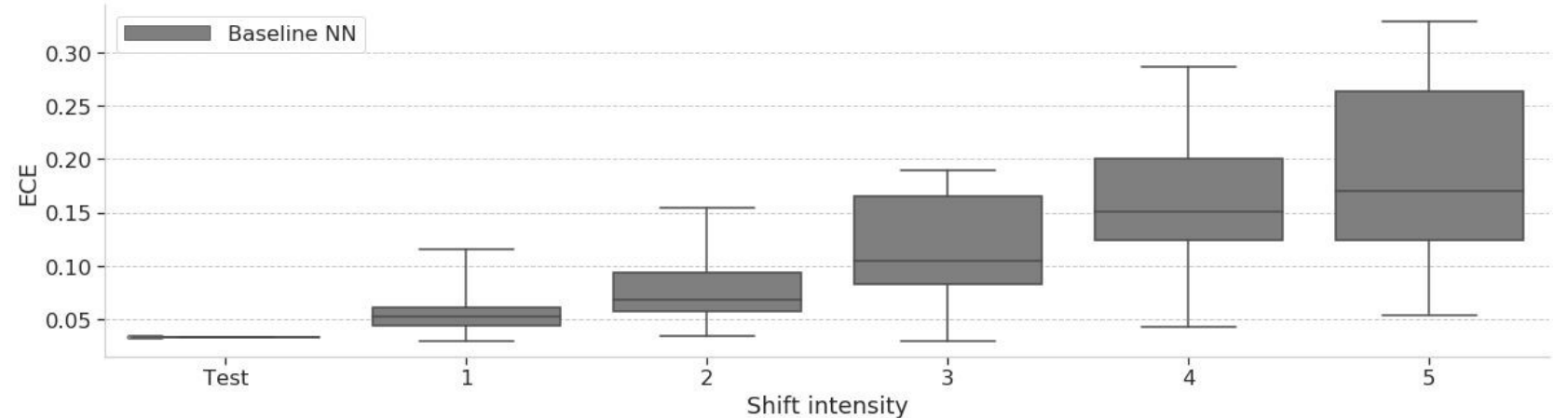
# Neural networks *do not know when they don't know*



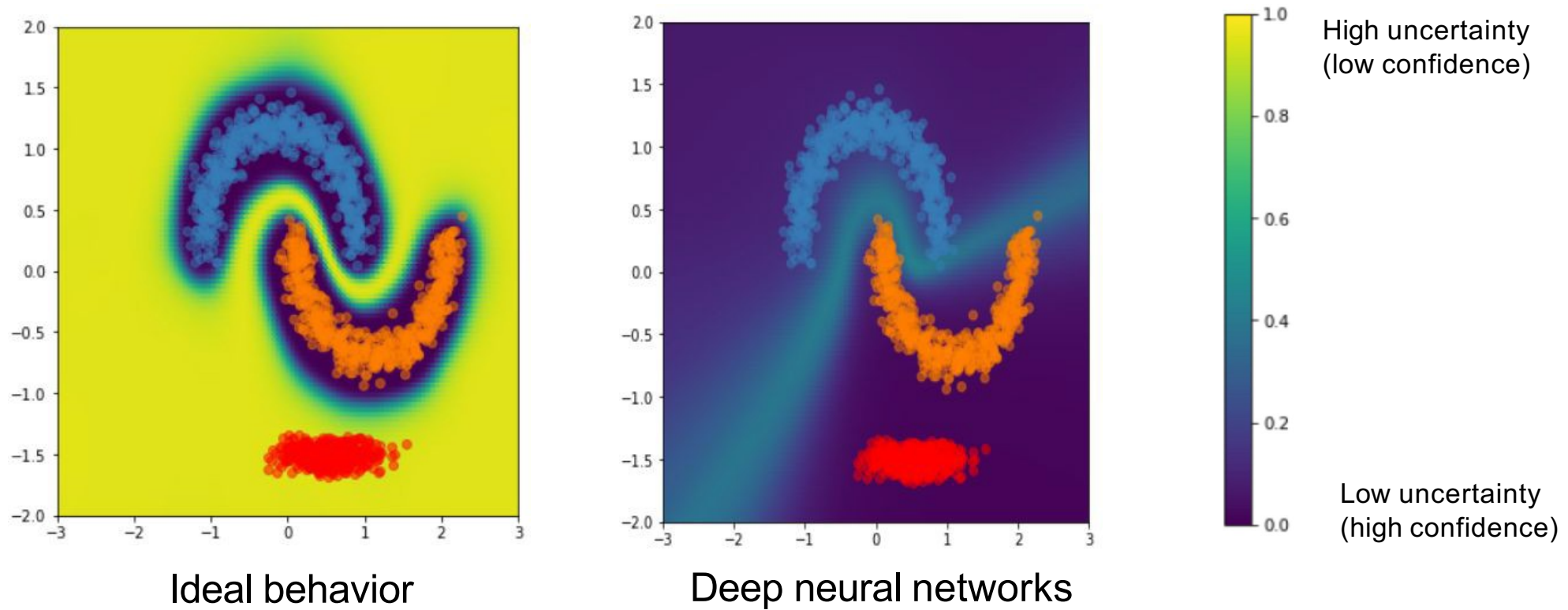
- **Accuracy drops** with increasing shift on Imagenet-C



- **Quality of uncertainty degrades with shift**  
-> “overconfident mistakes”



# Models assign high confidence predictions to OOD inputs

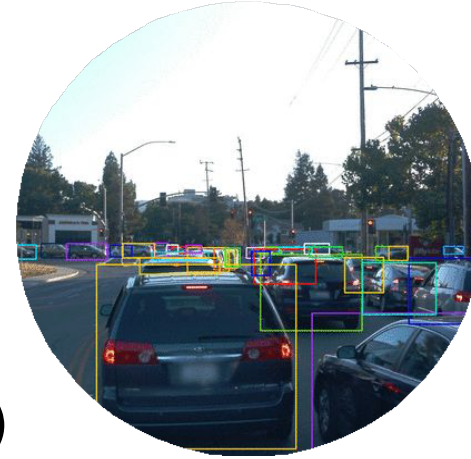


Trust model when  $x^*$  is close to  $p_{\text{TRAIN}}(x,y)$

# Self-driving cars

## Dataset shift:

- Time of day / Lighting
- Geographical location (City vs suburban)
- Changing conditions (Weather / Construction)



Daylight



Night



Weather



Construction



Downtown



Suburban

Image credit: Sun et al, [Waymo Open Dataset](#)



# Open Set Recognition

- Example: Classification of genomic sequences
- High accuracy on known classes is not sufficient
- Need to be able to detect inputs that do not belong to one of the known classes

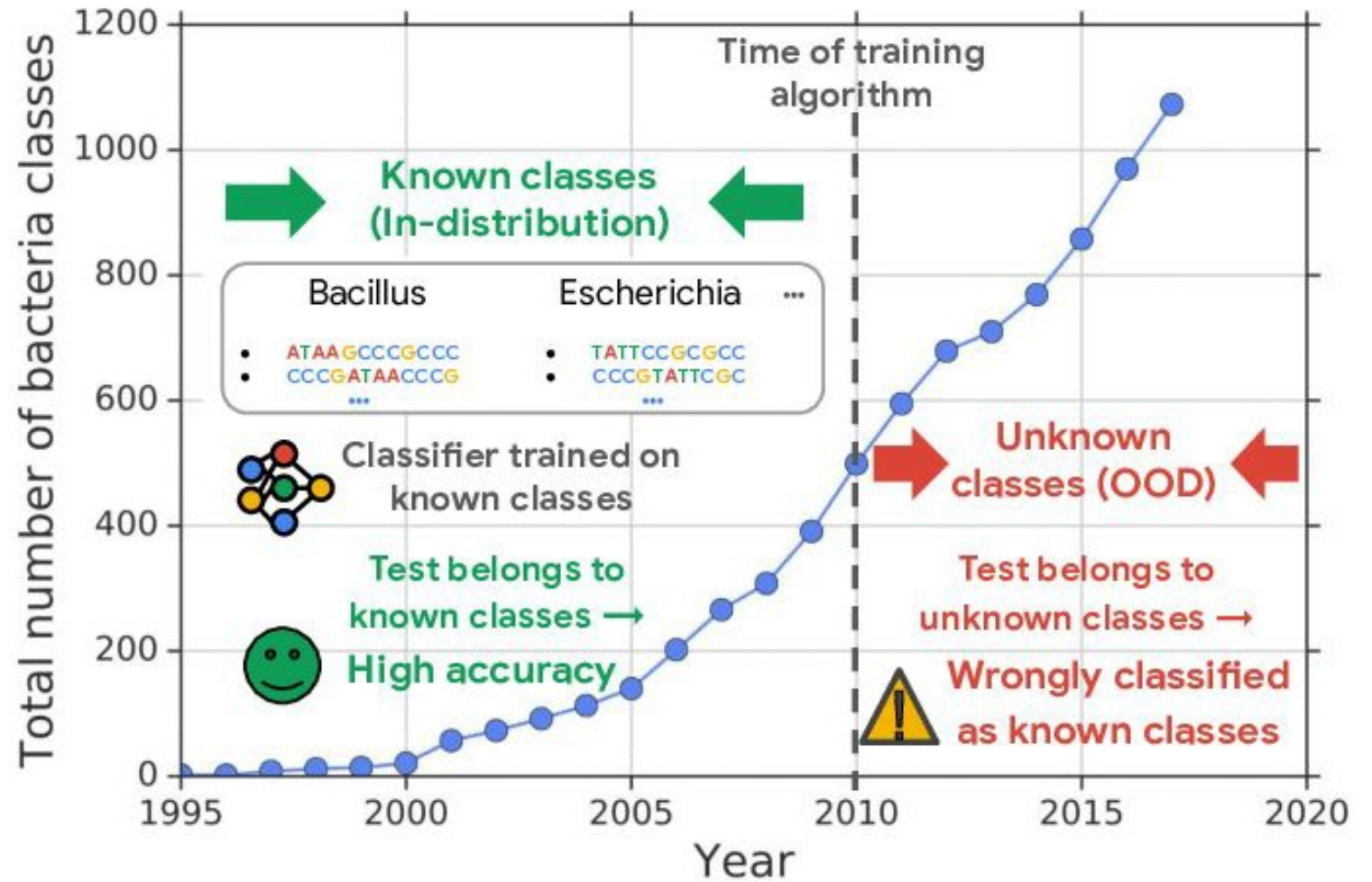
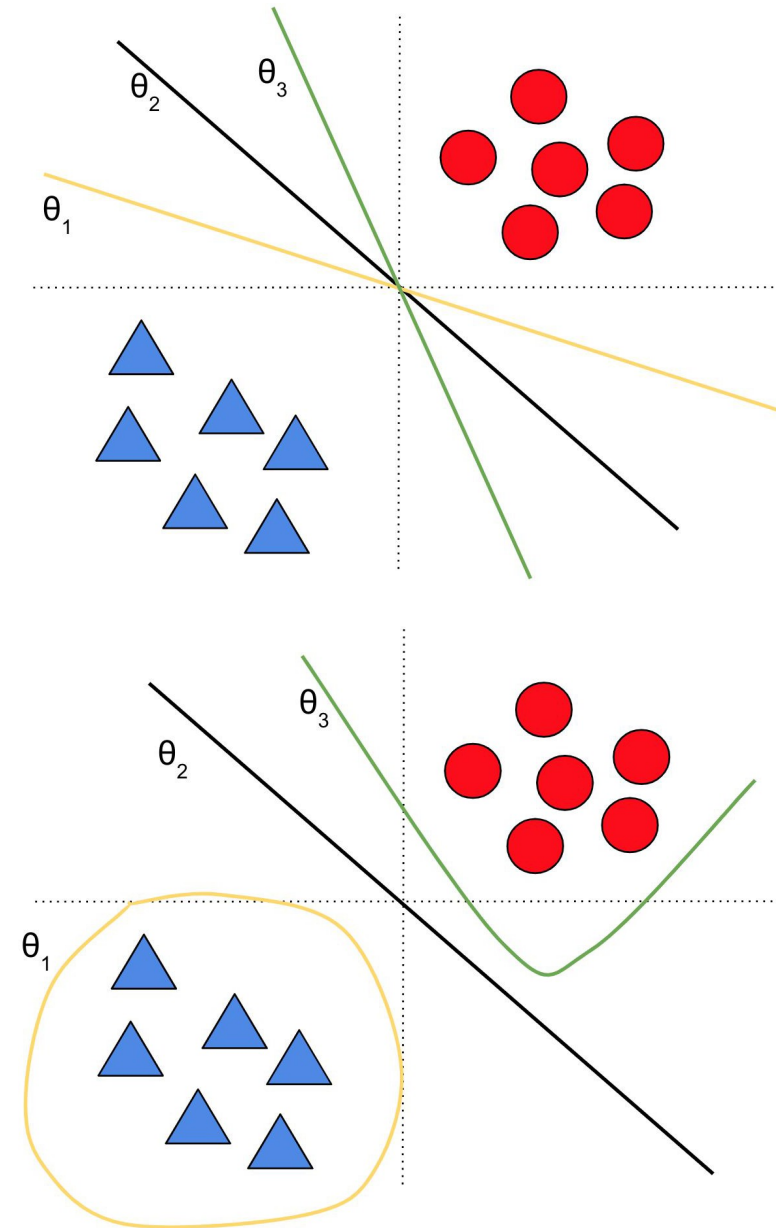


Image source: <https://ai.googleblog.com/2019/12/improving-out-of-distribution-detection.html>

# Sources of uncertainty: *Model uncertainty*

- Many models can fit the training data well
- Also known as *epistemic uncertainty*
- Model uncertainty is “**reducible**”
  - Vanishes in the limit of infinite data (subject to model identifiability)
- Models can be from same hypotheses class (e.g. linear classifiers in top figure) or belong to different hypotheses classes (bottom figure).



# Sources of uncertainty: *Data uncertainty*

- Labeling noise (ex: human disagreement)
- Measurement noise (ex: imprecise tools)
- *Missing* data (ex: partially observed features, unobserved confounders)
- Also known as *aleatoric uncertainty*
- Data uncertainty is “**irreducible\***”
  - Persists even in the limit of infinite data
  - \*Could be reduced with additional features/views

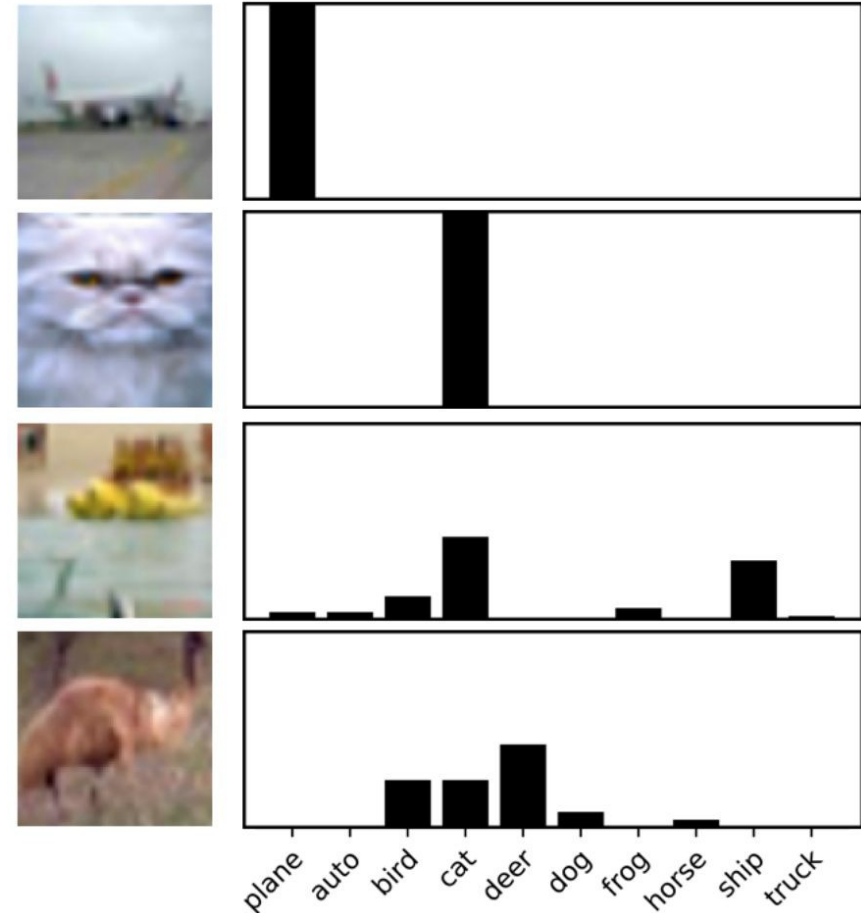


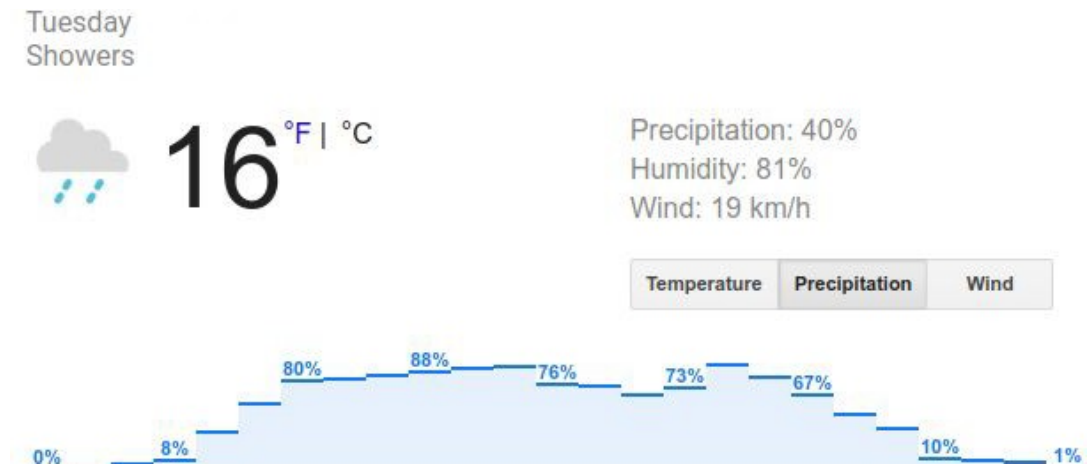
Image source: [Battleday et al. 2019](#) “Improving machine classification using human uncertainty measurements”

# How do we measure the quality of uncertainty?

$$\text{Calibration Error} = |\text{Confidence} - \text{Accuracy}|$$

Of all the days where the model predicted rain with 80% probability, what fraction did we observe rain?

- 80% implies perfect calibration
- Less than 80% implies model is overconfident
- Greater than 80% implies model is under-confident



# How do we measure the quality of uncertainty?

Expected Calibration Error [[Naeini+ 2015](#)]:

$$\text{ECE} = \sum_{b=1}^B \frac{n_b}{N} |\text{acc}(b) - \text{conf}(b)|$$

- Bin the probabilities into  $B$  bins.
- Compute the within-bin accuracy and within-bin predicted confidence.
- Average the calibration error across bins (weighted by number of points in each bin).

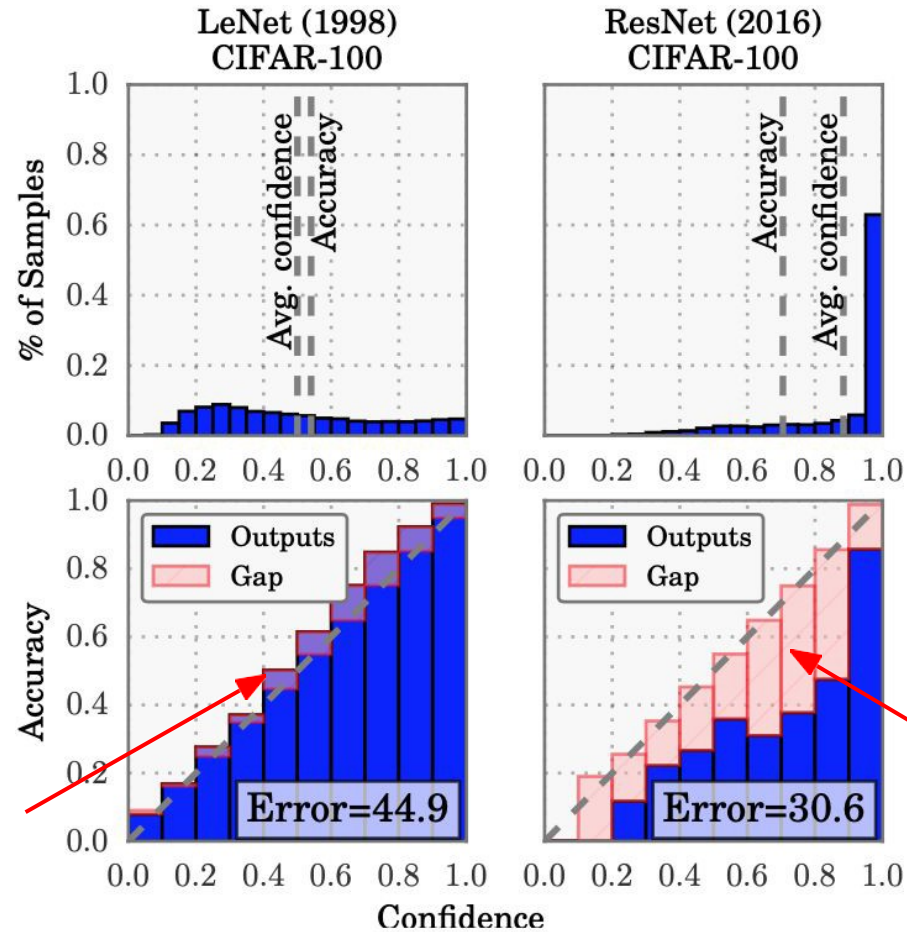
# How do we measure the quality of uncertainty?

Expected Calibration Error [Naeini+ 2015]:

$$\text{ECE} = \sum_{b=1}^B \frac{n_b}{N} |\text{acc}(b) - \text{conf}(b)|$$

*Confidence < Accuracy*

*=> Underconfident*



*Confidence > Accuracy*

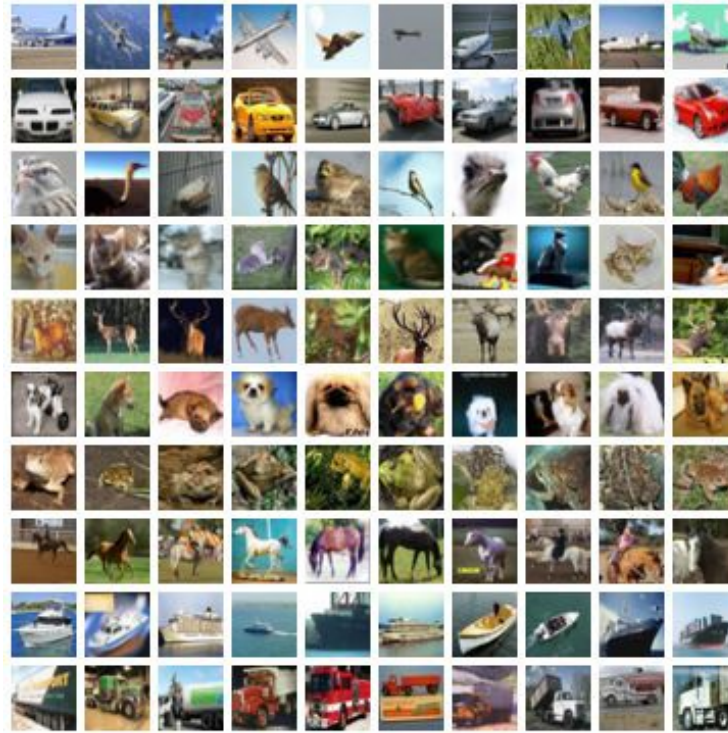
*=> Overconfident*

Image source: [Guo+ 2017](#) "On calibration of modern neural networks"

# How do we measure the quality of uncertainty, practically?

Evaluate model on **out-of-distribution (OOD) inputs** which do not belong to any of the existing classes

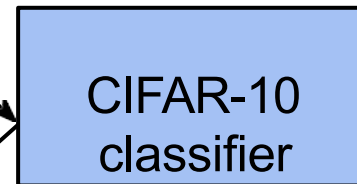
- Max confidence
- Entropy of  $p(y|x)$



CIFAR-10 (i.i.d test inputs)



SVHN (o.o.d test inputs)



Confidence on i.i.d inputs



Confidence on o.o.d inputs ?

# How do we measure the quality of robustness, practically?

Measure generalization to a *large collection of real-world shifts*. A large collection of tasks encourages *general robustness to shifts* (ex: [GLUE](#) for NLP).

- Novel textures in object recognition.
- Covariate shift (e.g. corruptions).
- Different sub-populations (e.g. geographical location).



Cartoon

Different renditions  
(ImageNet-R)



Nearby video frames  
(ImageNet-Vid-Robust, YTBB-Robust)



Multiple objects and poses  
(ObjectNet)

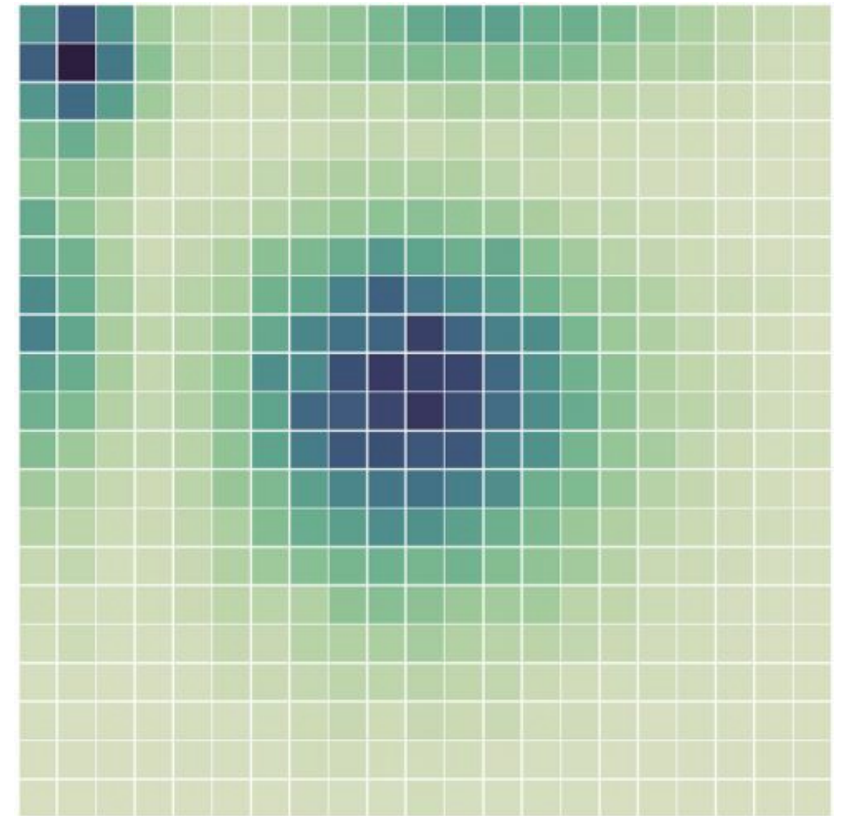


# Neural Networks with SGD

Nearly all models find a single setting of parameters to maximize the probability conditioned on data.

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{y}) \\ &= \arg \min_{\boldsymbol{\theta}} -\log p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}) - \log p(\boldsymbol{\theta})\end{aligned}$$

↑  
Data uncertainty



Special case: softmax cross entropy with L2 regularization. Optimize with SGD!

Image source: [Ranganath+ 2016](#)

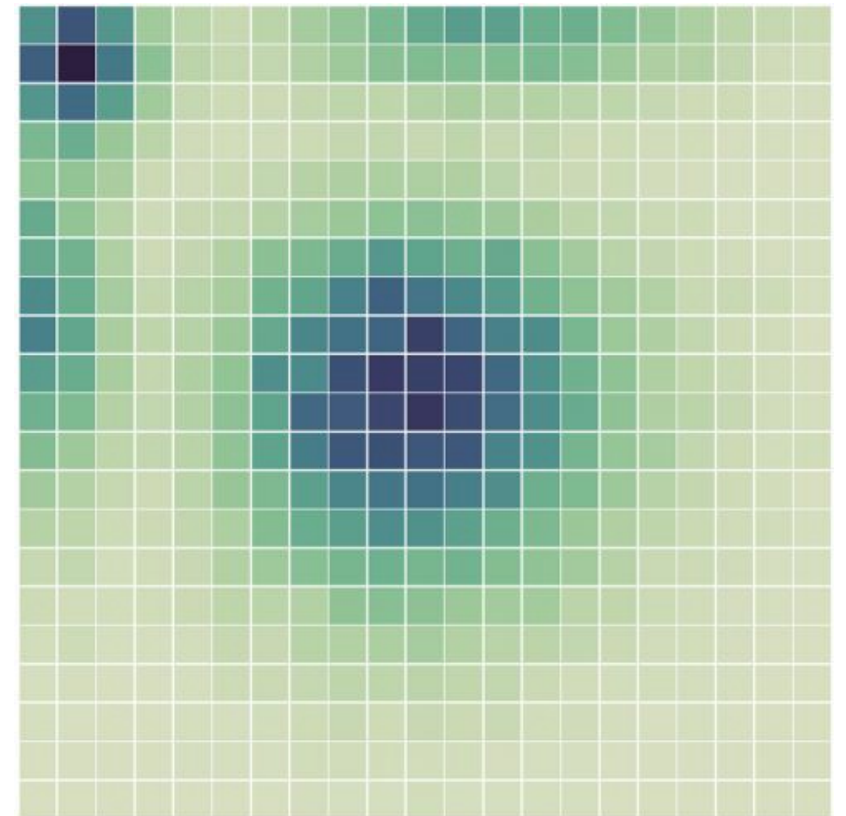
# A Simple Baseline for Improving Uncertainty Calibration

$$\theta^* = \arg \max_{\theta} p(\theta \mid \mathbf{x}, \mathbf{y})$$

Problem: results in just one prediction per example  
\*No model uncertainty\*

How do we get uncertainty?

- Probabilistic approach
  - Estimate a full distribution for  $p(\theta \mid \mathbf{x}, \mathbf{y})$
- Intuitive approach: Ensembling
  - Obtain multiple good settings for  $\theta^*$



# Ensemble Learning

- A prior distribution often involves the complication of approximate inference.
- *Ensemble learning* offers an alternative strategy to aggregate the predictions over a collection of models.
- Often winner of competitions!
- There are two considerations: the collection of models to ensemble; and the aggregation strategy.

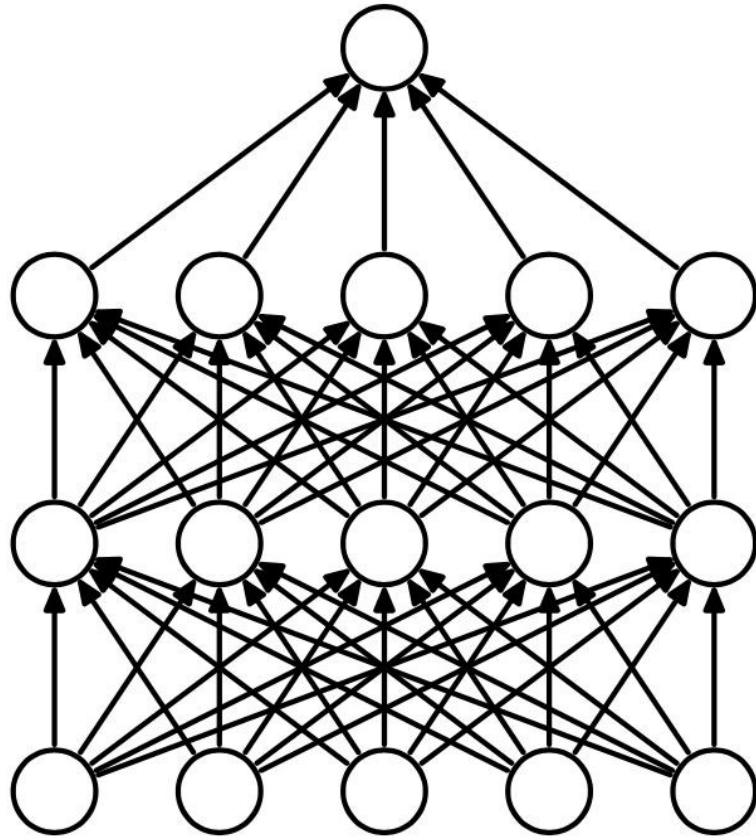
Popular approach is to average predictions of independently trained models, forming a mixture distribution.

$$p(\mathbf{y} | \mathbf{x}) = \frac{1}{K} \sum_{k=1}^K p(\mathbf{y} | \mathbf{x}, \boldsymbol{\theta}_k)$$

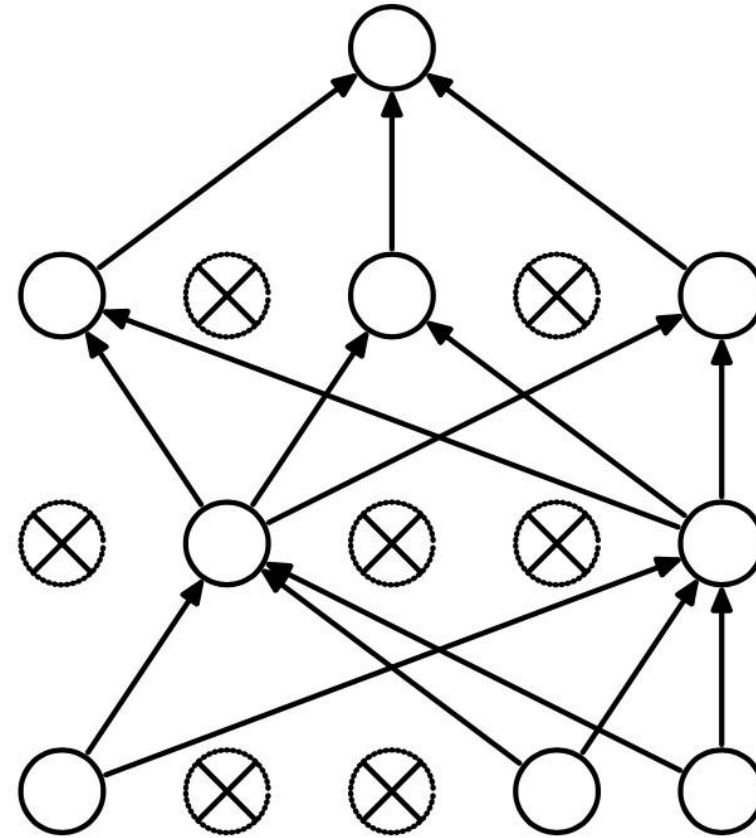
Many approaches exist: bagging, boosting, decision trees, stacking.

*...What this reminds you in neural networks?*

# An Old Friend Wears A New Hat: (Monte Carlo) Dropout!



(a) Standard Neural Net



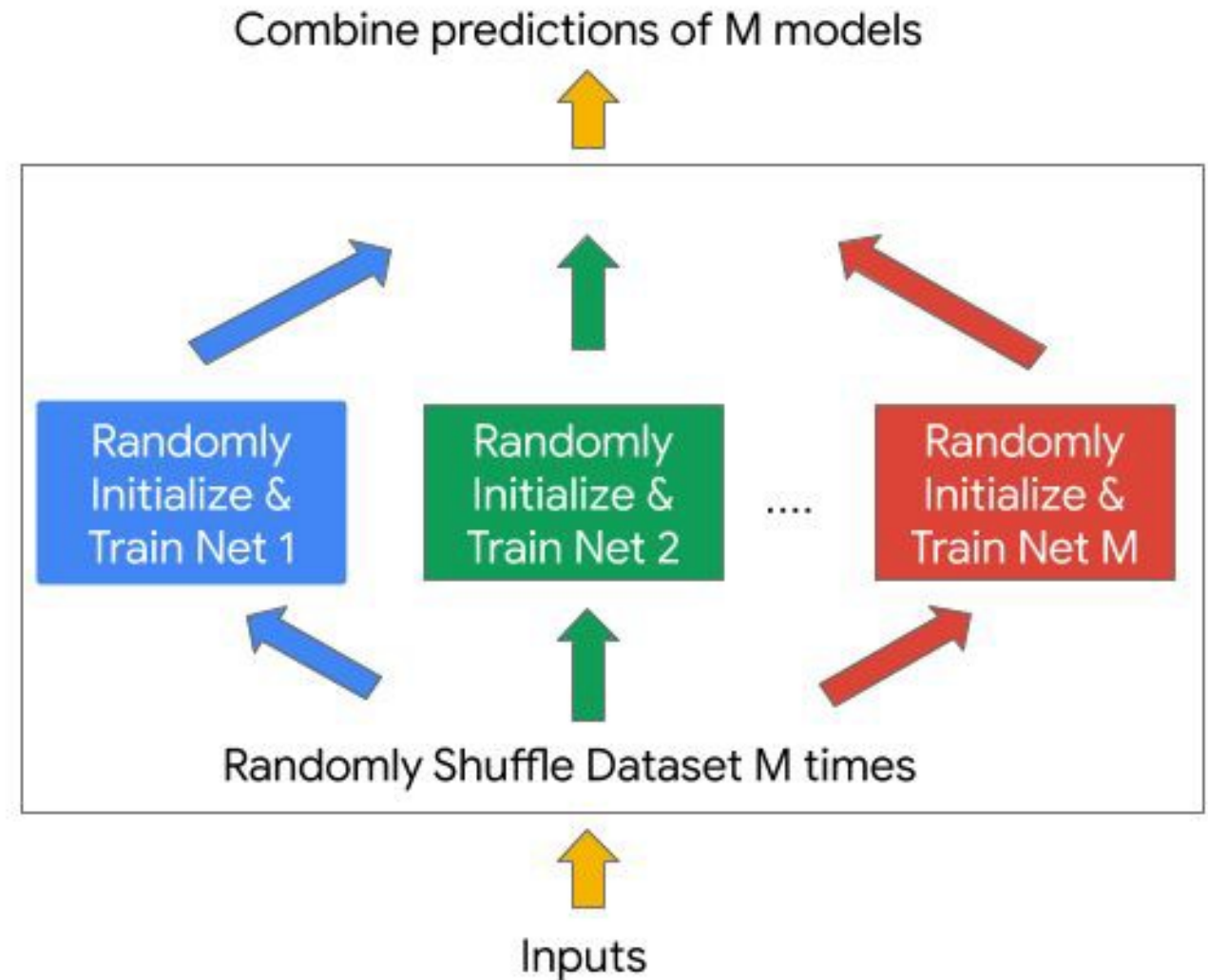
(b) After applying dropout.

Image source: Dropout: A Simple Way to Prevent Neural Networks from Overfitting

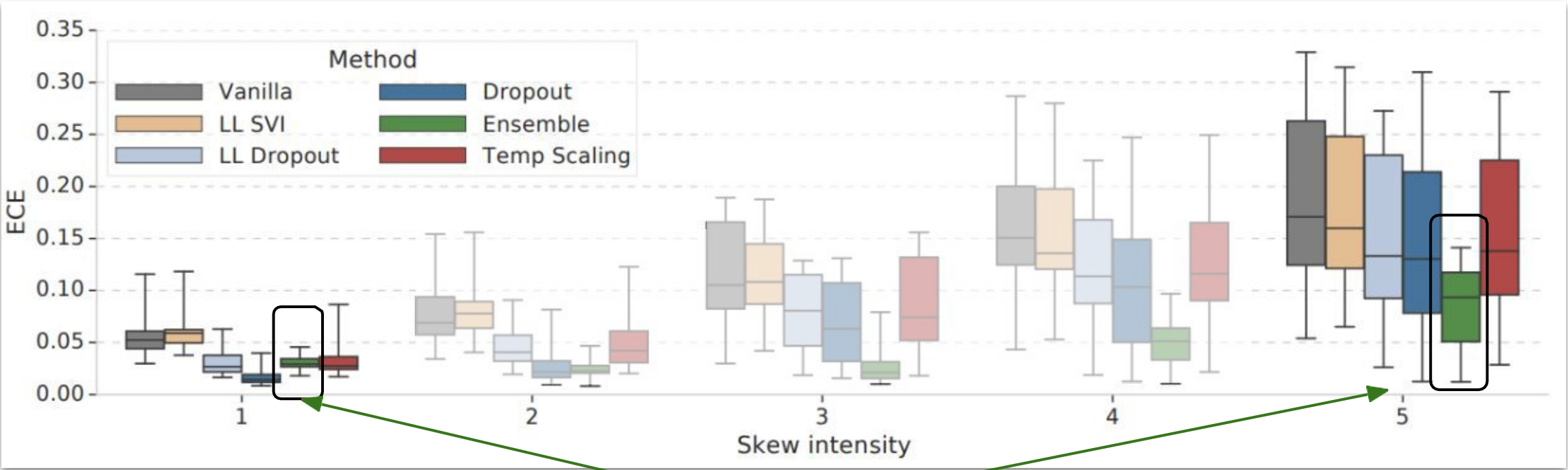
# Simple Baseline: Deep Ensembles

Idea: Just re-run standard SGD training but with different random seeds and average the predictions

- A well-known trick for getting better accuracy and Kaggle scores
- **Beyond accuracy – it is good for robustness and uncertainty too!!**
- The **mean** of predictions is often more accurate, and the **variance** of those predictions reflect “confidence”
- We rely on the facts that the loss landscape is non-convex and SGD has noise



# Deep Ensembles work surprisingly well in practice



*Deep Ensembles are consistently among the best performing methods, especially under dataset shift*

Dataset	Model	Uncalibrated	Hist. Binning	Isotonic	BBQ	Temp. Scaling	Vector Scaling	Matrix Scaling
Birds	ResNet 50	0.10%	1.24%	5.22%	1.12%	1.85%	2.00%	21.13%
Cars								0.5%
CIFAR-10								1.0%
CIFAR-10								0.72%
CIFAR-10								0.72%
CIFAR-10								0.41%
CIFAR-10								0.16%
CIFAR-100								5.49%
CIFAR-100								0.09%
CIFAR-100								4.44%
CIFAR-100								1.87%
CIFAR-100								3.24%
ImageNet								-
ImageNet								-
SVHN								0.17%
20 News								0.1%
Reuters								0.58%
SST Binary								0.84%
SST Fine Grai								0.39%



Table 1. ECE (The number for

methods.

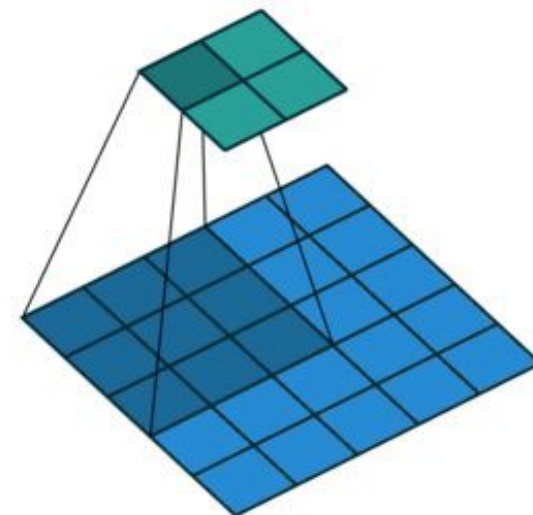
Softmax:  $\sigma_{SM}(\mathbf{z}_i)^{(k)} = \frac{\exp(z_i^{(k)})}{\sum_{j=1}^K \exp(z_i^{(j)})}$ ,  $\hat{p}_i = \max_k \sigma_{SM}(\mathbf{z}_i)^{(k)}$ . Temperature re-scaling (beat them all!):  $\hat{q}_i = \max_k \sigma_{SM}(\mathbf{z}_i/T)^{(k)}$ .

# Inductive Priors & Knowledge: Another Powerful Tool for Uncertainty & Robustness

What about inductive biases to assist OOD?

Image source: [Dumoulin & Visin 2016](#)

- Hypothesis: *“Representations should be invariant with respect to dataset shift.”*
- **Data augmentation** extends the dataset in order to encourage invariances.
- More examples: **contrastive learning, equivariant architectures.**

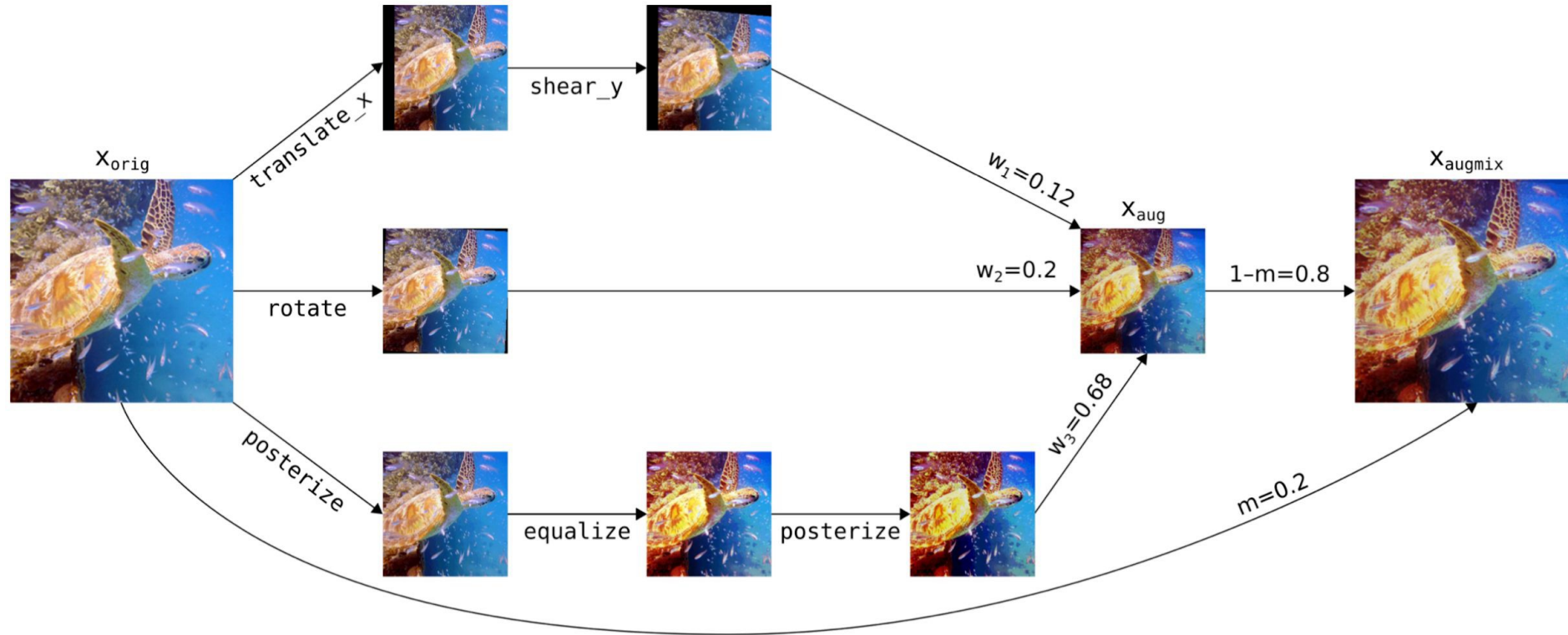


Data augmentation requires two considerations:

1. Set of **base** augmentation operations. (Ex: color distortions, word substitution)
2. **Combination** strategy (Ex: Sequence of K randomly selected ops.)

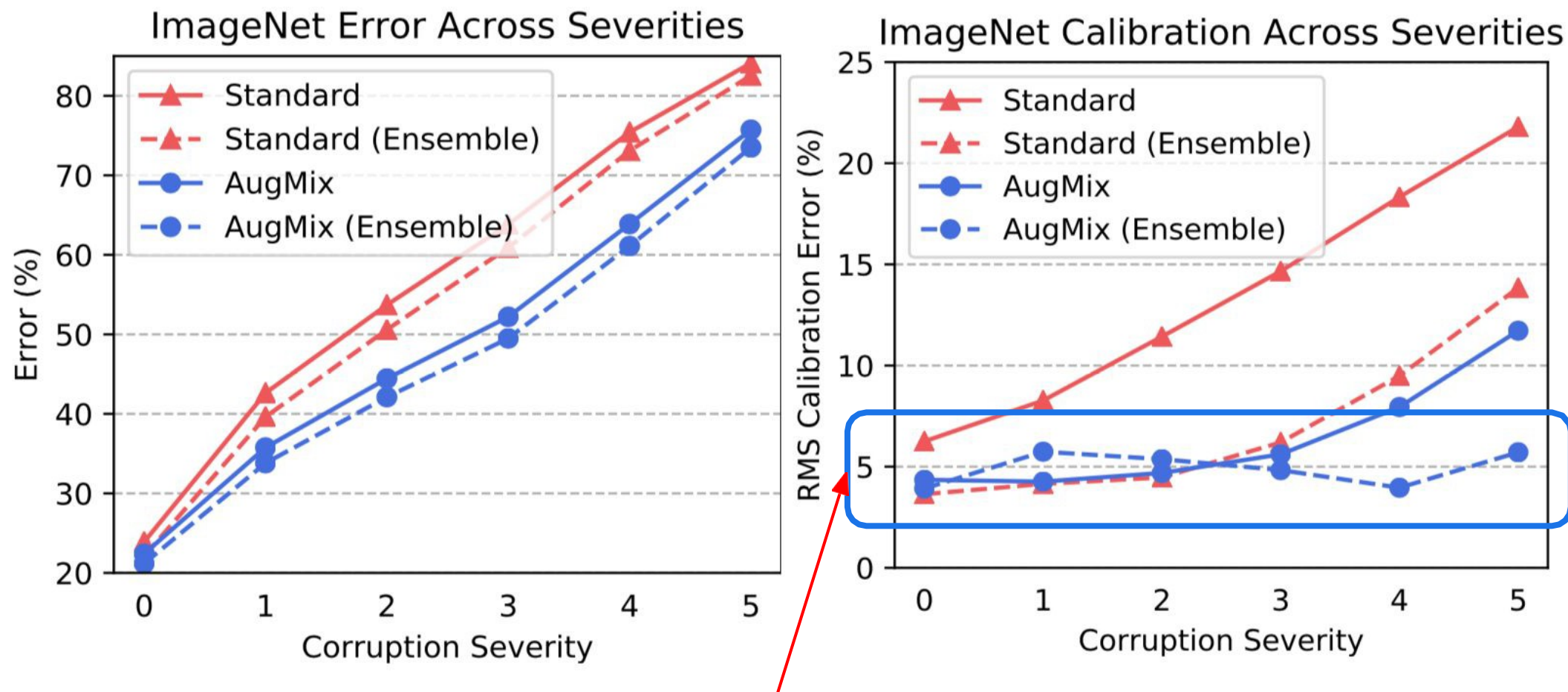


# Composing a set of base augmentations



Composing base operations and ‘mixing’ them can improve accuracy and calibration under shift.

# AugMix improves accuracy & calibration under shift



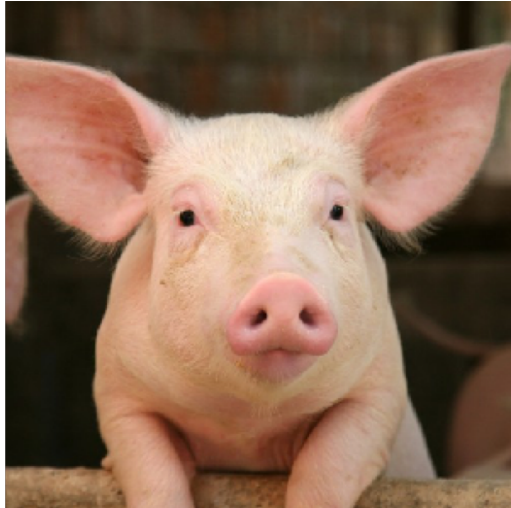
Data augmentation can provide complementary benefits to marginalization.

# Takeaways

- Uncertainty & robustness are critical problems in AI and machine learning.
- Benchmark models with calibration error and a large collection of OOD shifts.
- Probabilistic ML, ensemble learning, and optimization provide a foundation.
- The best methods: ensemble multiple predictions; imposing priors and inductive biases; and “lower your temperature” when using softmax
- Many future progress are expected – a key knob to make ML “real”

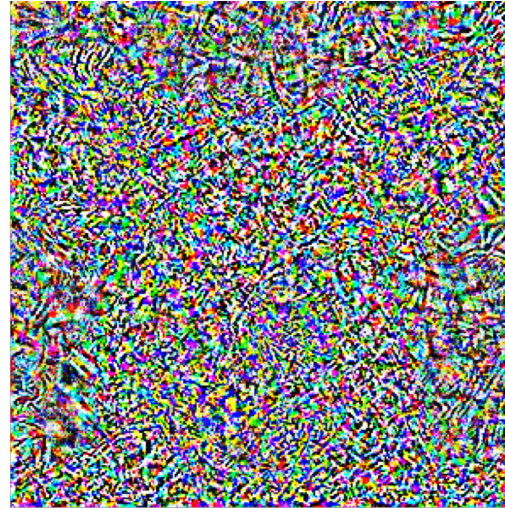
# ML Predictions Are (Mostly) Accurate but Brittle

“pig” (91%)



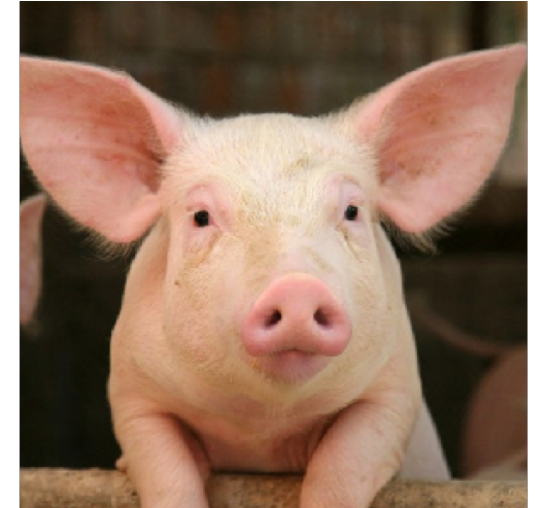
+ 0.005 x

noise (NOT random)



=

“airliner” (99%)



[Szegedy Zaremba Sutskever Bruna Erhan Goodfellow Fergus 2013]

[Biggio Corona Maiorca Nelson Srndic Laskov Giacinto Roli 2013]

**But also:** [Dalvi Domingos Mausam Sanghai Verma 2004][Lowd Meek 2005]

[Globerson Roweis 2006][Kolcz Teo 2009][Barreno Nelson Rubinstein Joseph Tygar 2010]

[Biggio Fumera Roli 2010][Biggio Fumera Roli 2014][Srndic Laskov 2013]

# Three commandments of Secure/Safe ML

I. Thou

(because

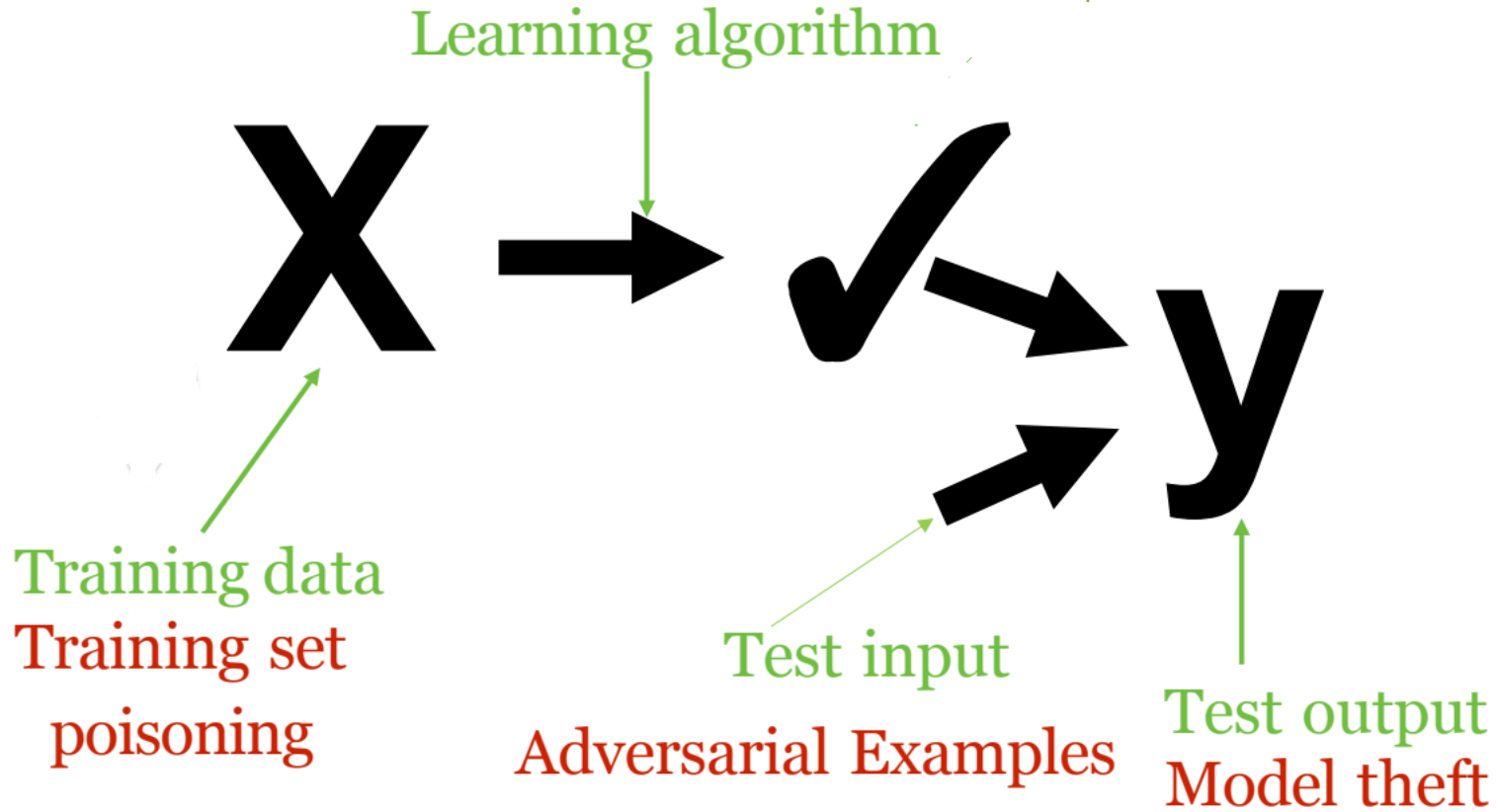
II. Thou

output

(because

III. Thou

(because of adversarial examples)



erve its

r model

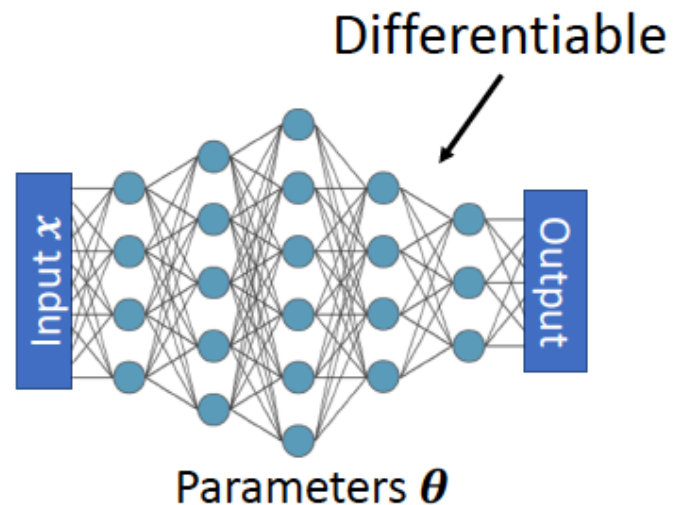
# Where Do Adversarial Examples Come From?

To get an adv. example

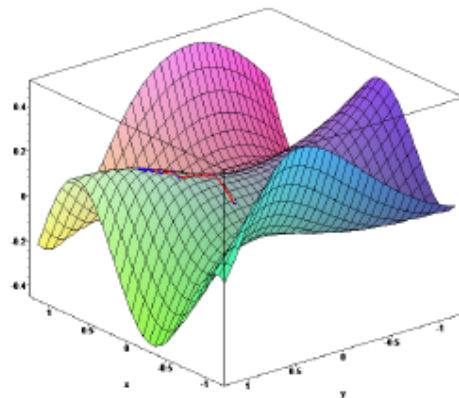
~~Goal of training:~~

Model Parameters    Input    Correct Label

$$\min_{\theta} \text{loss}(\theta, x, y)$$



Can use gradient descent method to find good  $\theta$

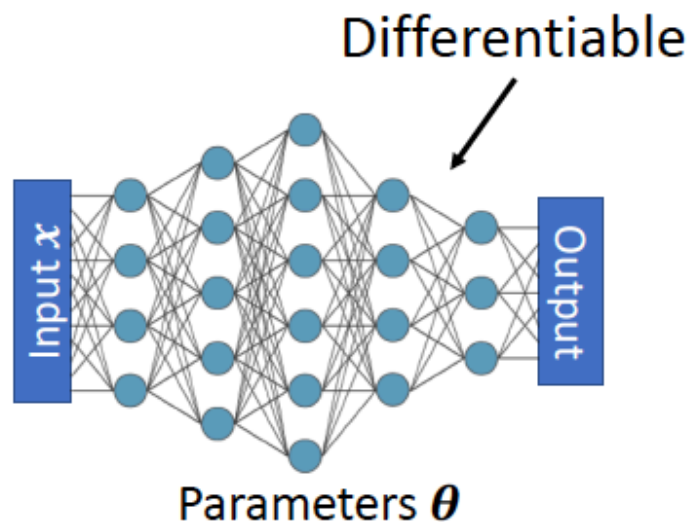


# Where Do Adversarial Examples Come From?

To get an adv. example

~~Goal of training:~~

$$\max_{\delta} \text{loss}(\theta, x + \delta, y)$$

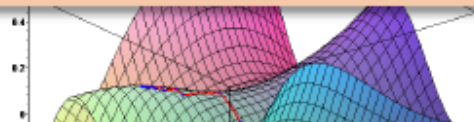


Which  $\delta$  are allowed?

**Examples:**  $\delta$  that is small wrt

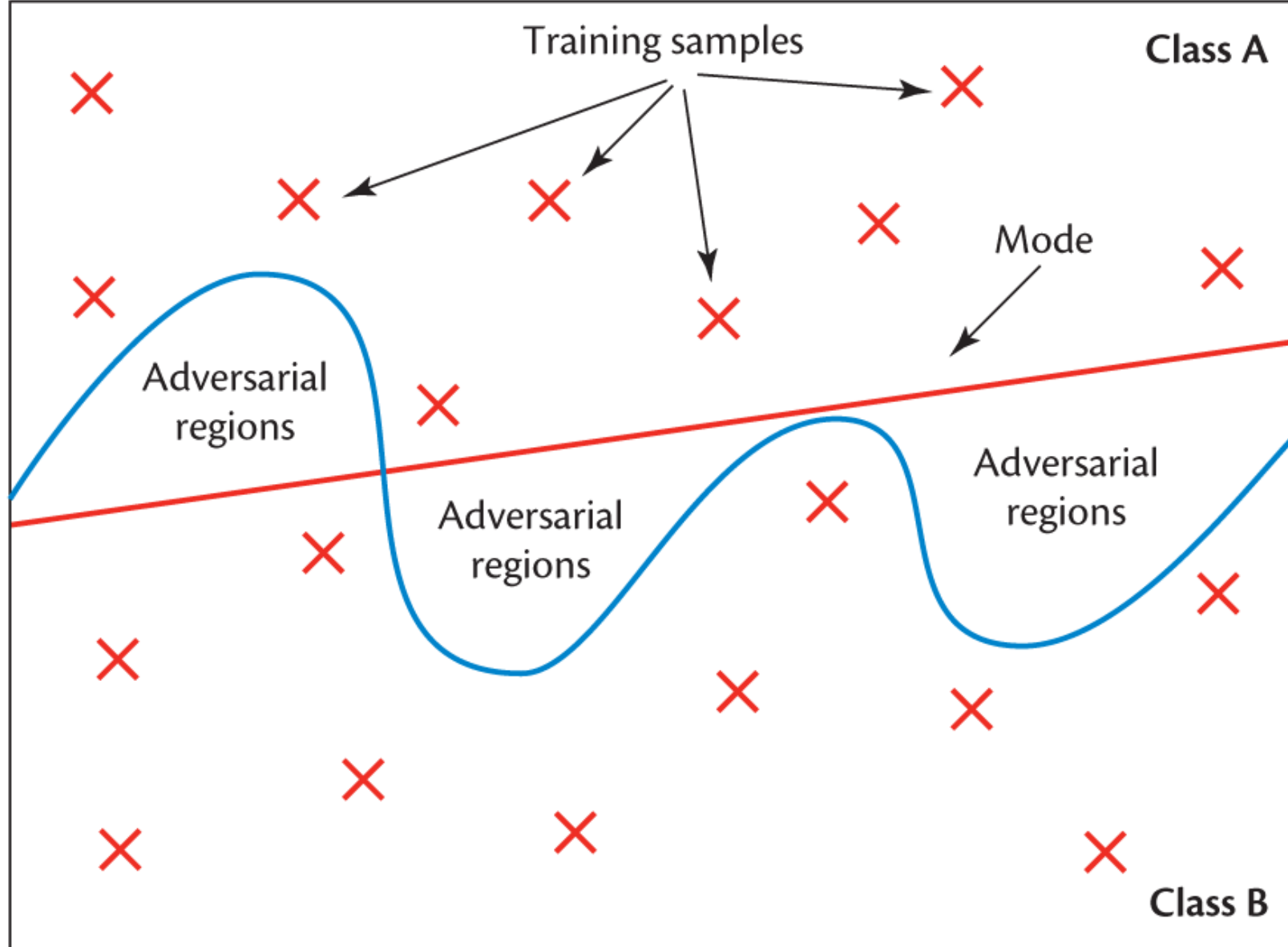
- $\ell_p$ -norm
- Rotation and/or translation
- VGG feature perturbation
- (add the perturbation you need here)

Can use gradient descent  
This is an important question  
(that we put aside)



**Still:** We have to confront  
(small)  $\ell_p$ -norm perturbations

# A Possible By-Product of ML Bias-Variance Trade-Off





# Towards ML Models that Are Adv. Robust

[M Makelov Schmidt Tsipras Vladu 2018]

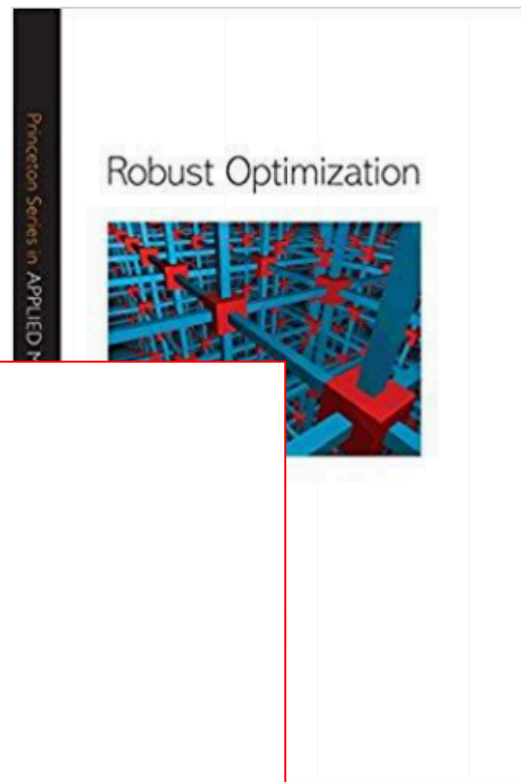
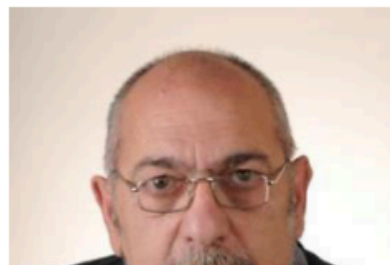
**Key observation:** Lack of adv. robustness is **NOT** at odds with what we currently want our ML models to achieve

~~Standard~~ generalization:  $\mathbb{E}_{(x,y) \sim D} [\max_{\delta \in \Delta} \text{loss}(\theta, x + \delta, y)]$

Adversarially robust

**But:** Adversarial noise is a “needle in a haystack”

# Robust Objectives



- Use the following

- $\min_w E$
- Outer minimization
- Inner maximization

## Part II: training a robust classifier

$$\min_{\theta} \sum_{x, y \in S} \max_{\delta \in \Delta} \text{Loss}(x + \delta, y; \theta)$$

Part I: creating an adversarial example  
(or ensuring one does not exist)

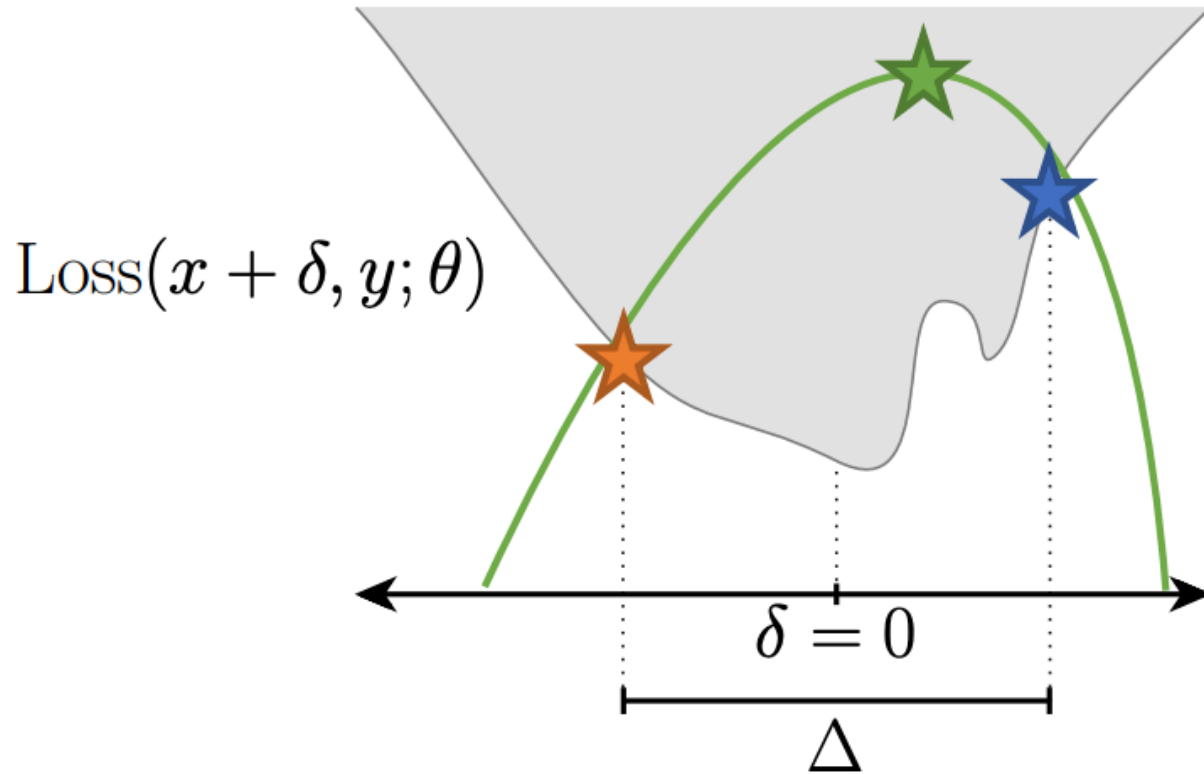
- A. Madry, *Adversarial*  
Learning M

- A. Sinha, H. Namkoong, and J. Duchi. Certifying some Distributional Robustness with Principled Adversarial Training. ICLR 2018

Deep

# The inner maximization problem

How do we solve the optimization?



$$\max_{\delta \in \Delta} \text{Loss}(x + \delta, y; \theta)$$

1. Local search (lower bound on objective)
2. Combinatorial optimization (exactly solve objective)
3. Convex relaxation (upper bound on objective)

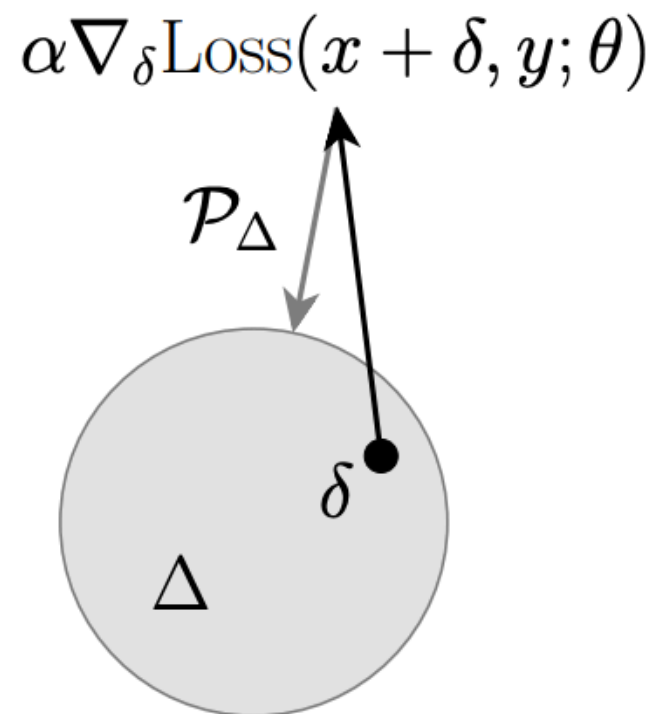
# Projected gradient descent

Recall we are optimizing

$$\max_{\delta \in \Delta} \text{Loss}(x + \delta, y; \theta)$$

We can employ a projected gradient descent method, take gradient step and project back into feasible set  $\Delta$

$$\delta := \mathcal{P}_{\Delta}[\delta + \nabla_{\delta} \text{Loss}(x + \delta, y; \theta)]$$



# The Fast Gradient Sign Method (FGSM)

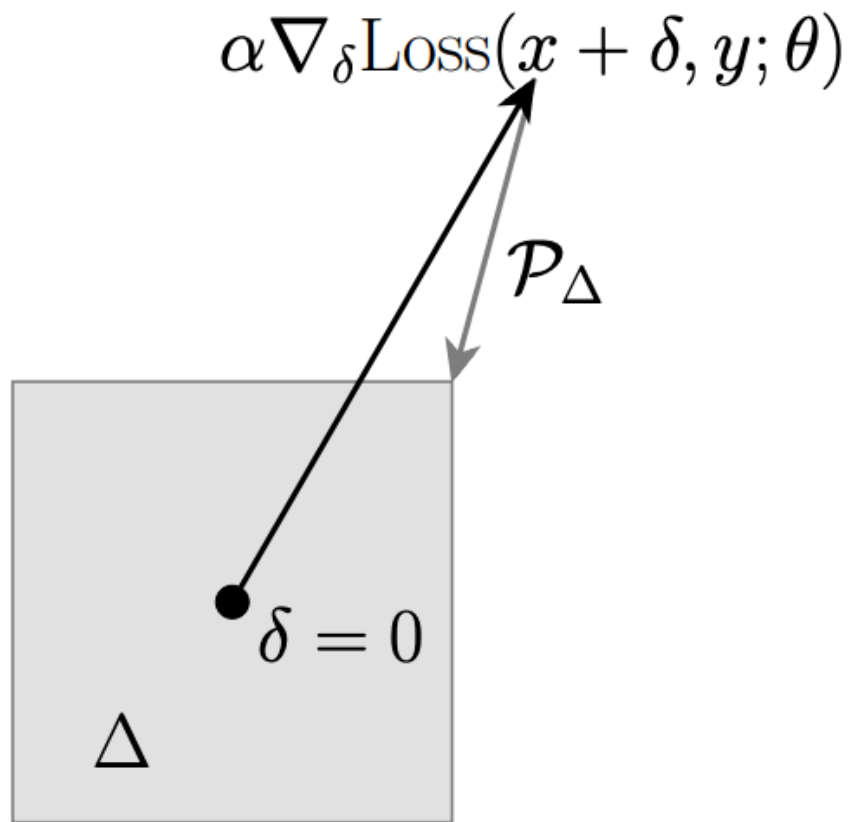
To be more concrete, take  $\Delta$  to be the  $\ell_\infty$  ball,  $\Delta = \{\delta: \|\delta\|_\infty \leq \epsilon\}$ , so projection takes the form

$$P_\Delta(\delta) = \text{Clip}(\delta, [-\epsilon, \epsilon])$$

As  $\alpha \rightarrow \infty$ , we always reach “corner” of the box, called fast gradient sign method (FGSM)

[Goodfellow et al., 2014]

$$\delta = \epsilon \cdot \text{sign}(\nabla_\delta \text{Loss}(x + \delta, y; \theta))$$



# Targeted attacks

Also possible to explicitly try to change label to a *particular* class

$$\max_{\delta \in \Delta} \left( \text{Loss}(x + \delta, y; \theta) - \text{Loss}(x + \delta, y_{\text{targ}}; \theta) \right)$$

Consider multi-class cross entropy loss

$$\text{Loss}(x + \delta, y; \theta) = \log \sum_i \exp h_{\theta}(x + \delta)_i - h_{\theta}(x)_y$$

Then note that above problem simplifies to

$$\max_{\delta \in \Delta} \left( h_{\theta}(x)_{y_{\text{targ}}} - h_{\theta}(x)_y \right)$$

# The outer minimization problem

**Inner maximization:**

$$\max_{\delta \in \Delta} \text{Loss}(x + \delta, y; \theta)$$



**Outer minimization:**

$$\min_{\theta} \sum_{x, y \in S} \max_{\delta \in \Delta} \text{Loss}(x + \delta, y; \theta)$$

1. Local search (lower bound on objective)
2. Combinatorial optimization (exactly solve objective)
3. Convex relaxation (upper bound on objective)

1. Adversarial training
3. Provably robust training

# Danskin's Theorem

A fundamental result in optimization:

$$\nabla_{\theta} \max_{\delta \in \Delta} \text{Loss}(x + \delta, y; \theta) = \nabla_{\theta} \text{Loss}(x + \delta^*, y; \theta)$$

where  $\delta^* = \max_{\delta \in \Delta} \text{Loss}(x + \delta, y; \theta)$

Seems “obvious,” but it is a very subtle result; means we can optimize through the max by just finding it's maximizing value

Note however, it *only* applies when max is performed exactly



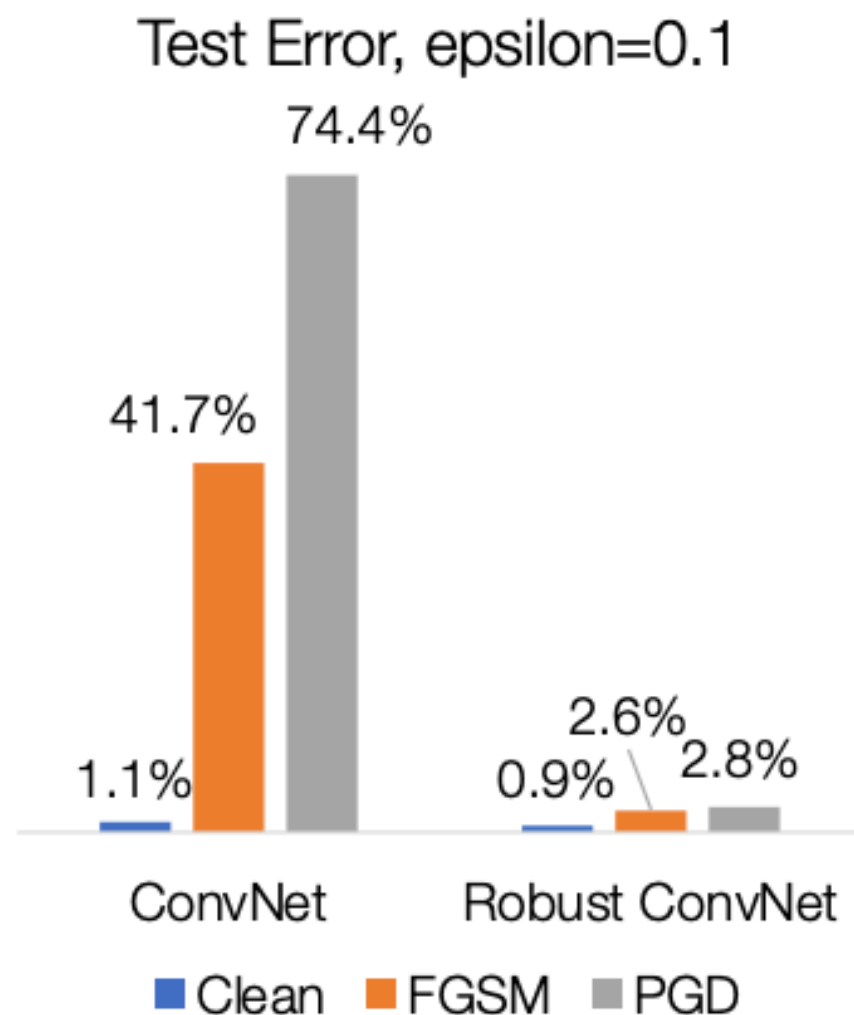
# Adversarial training [Goodfellow et al., 2014]

Repeat

1. Select minibatch  $B$
2. For each  $(x, y) \in B$ , compute adversarial example  $\delta^*(x)$
3. Update parameters

$$\theta := \theta - \frac{\alpha}{|B|} \sum_{x, y \in B} \nabla_{\theta} \text{Loss}(x + \delta^*(x), y; \theta)$$

Common to also mix robust/standard updates (not done in our case)



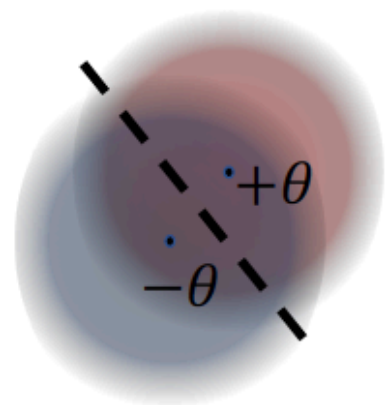
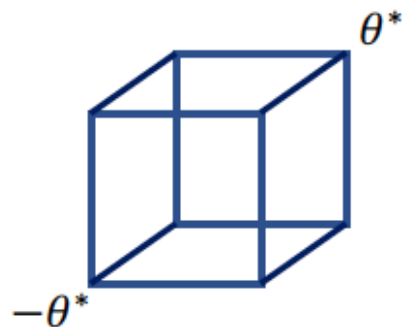
# Adv. Robust Generalization Needs More Data

Theorem [Schmidt Santurkar Tsipras Talwar **M** 2018]:

Sample complexity of adv. robust generalization can be **significantly larger** than that of “standard” generalization

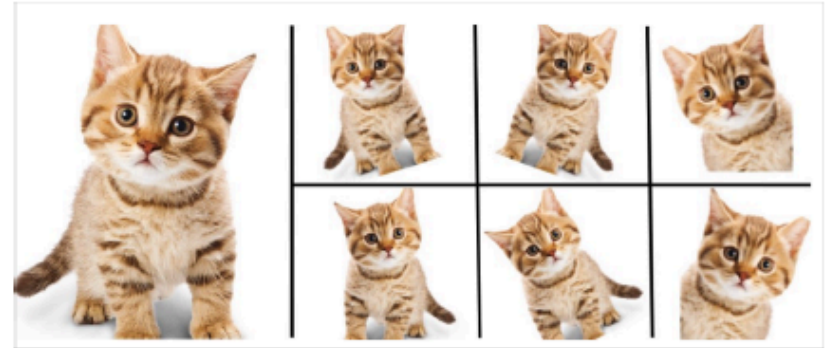
**Specifically:** There exists a  $d$ -dimensional distribution  $\mathbf{D}$  s.t.:

- A **single** sample is enough to get an **accurate** classifier ( $P[\text{correct}] > 0.99$ )
- **But:** Need  $\Omega(\sqrt{d})$  samples for better-than-chance **robust** classifier



# Does Being Robust Help “Standard” Generalization?

**Data augmentation:** An effective technique to improve “standard” generalization



Adversarial training

=

An “ultimate” version of data augmentation?

(since we train on the “most confusing” version of the training set)

Does adversarial training always improve  
“standard” generalization?

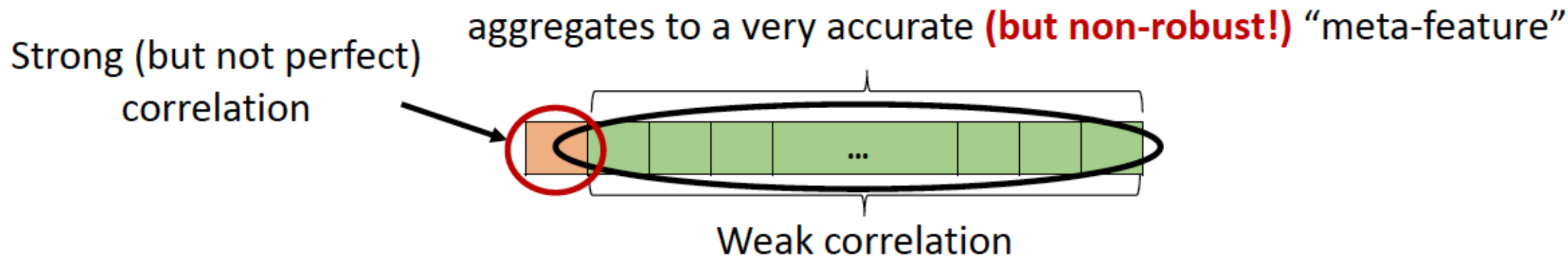
# Does Being Robust Help “Standard” Generalization?

Theorem [Tsipras Santurkar Engstrom Turner M 2018]:

No “free lunch”: can exist a trade-off between accuracy and robustness

## Basic intuition:

- In standard training, **all correlation is good correlation**
- If we want robustness, **must avoid** weakly correlated features

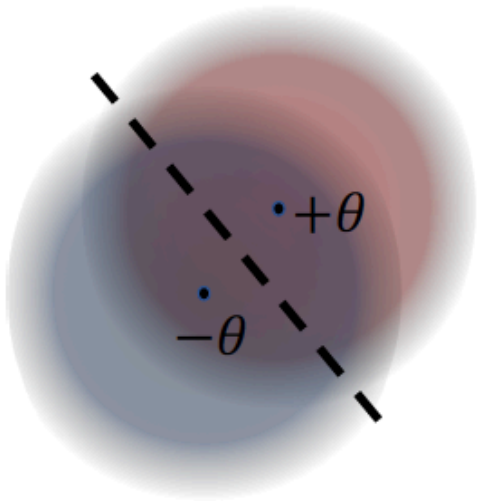
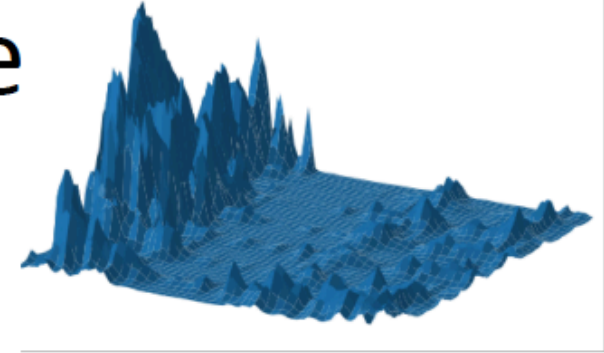


**Standard training:** use all of features, maximize accuracy

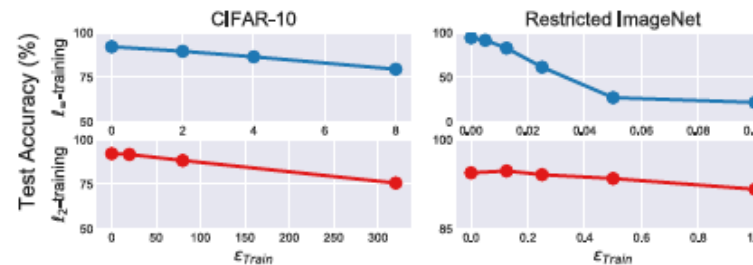
**Adversarial training:** use only single robust feature **(at the expense of accuracy)**

# Adversarial Robustness is Not Free

→ Optimization during training more difficult and models need to be larger



→ More training data might be required  
[Schmidt Santurkar Tsipras Talwar **M** 2018]



→ Might need to lose on “standard” measures of performance  
[Tsipras Santurkar Engstrom Turner **M** 2018] (Also see: [Bubeck Price Razenshteyn 2018])

-> Other Difficulties such as Robust Overfitting (ICML 2020) etc.

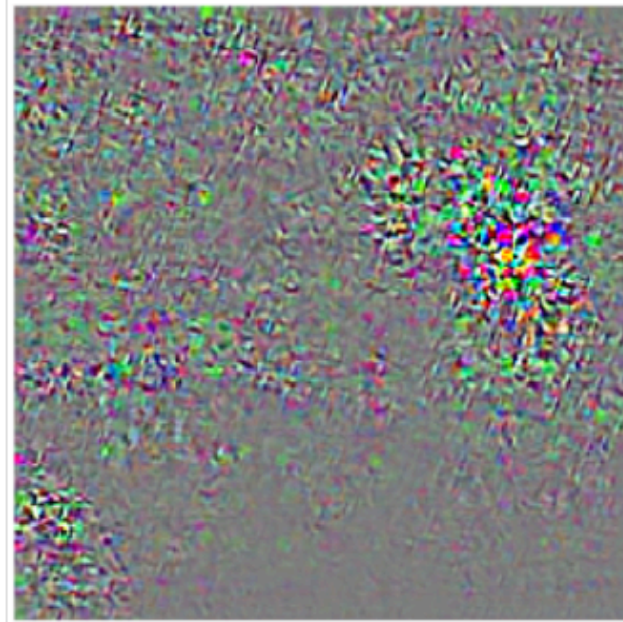
# But There Are (Unexpected?) Benefits Too

[Tsipras Santurkar Engstrom Turner M 2018]

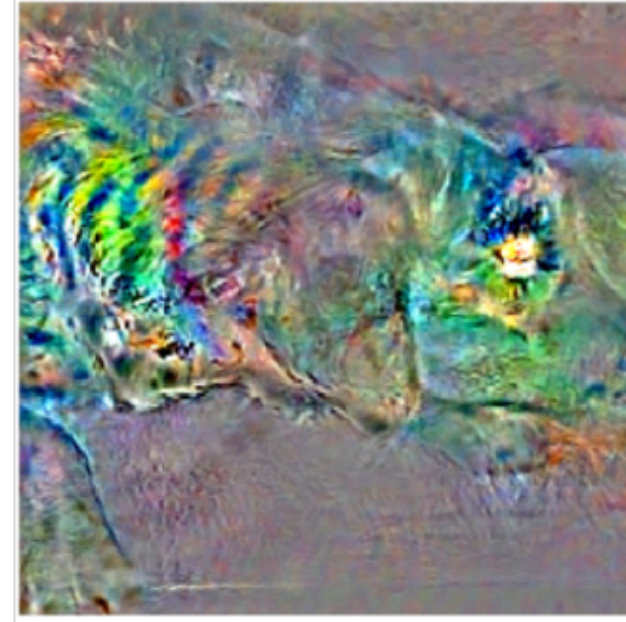
Models become more **semantically meaningful**



Input

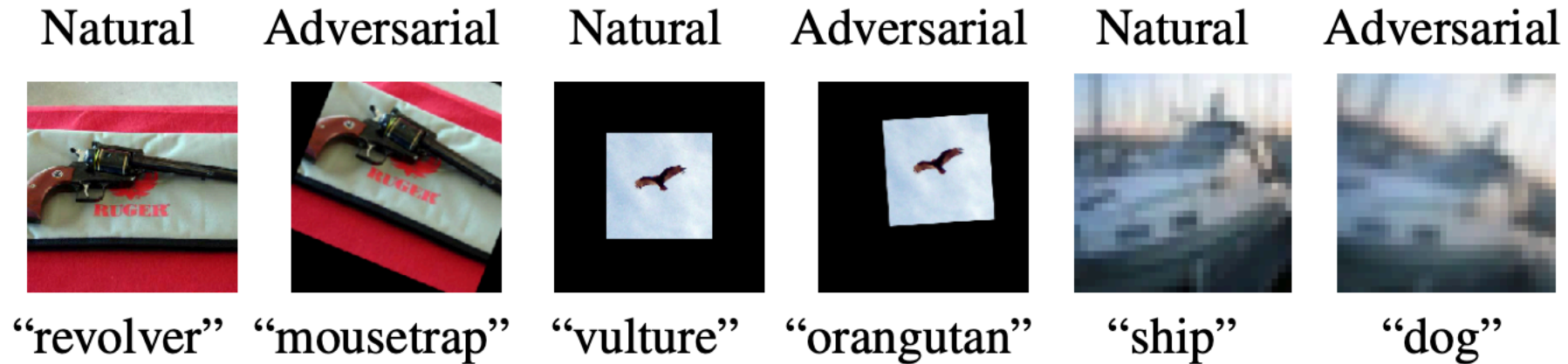


Gradient of  
standard model



Gradient of  
**adv. robust** model

# Adversarial Examples Beyond Pixel Perturbations ...



## A ROTATION AND A TRANSLATION SUFFICE: FOOLING CNNs WITH SIMPLE TRANSFORMATIONS

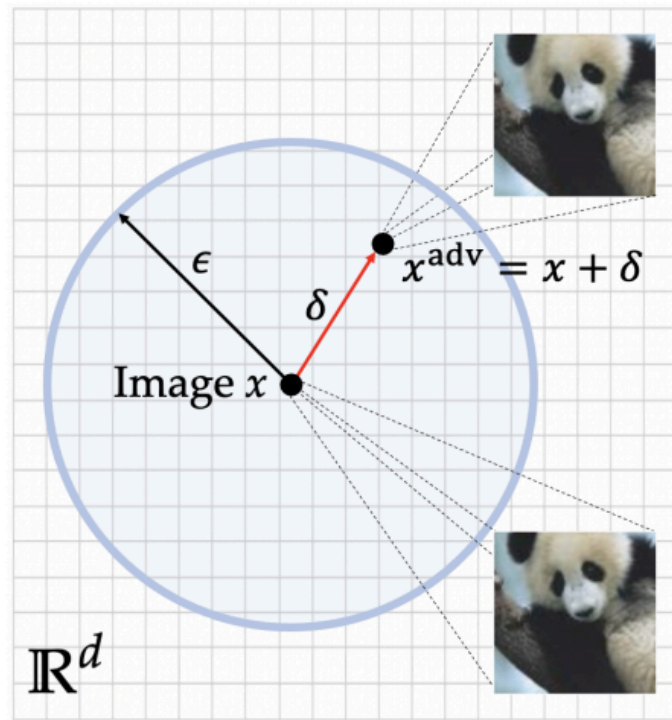
Logan Engstrom, Ludwig Schmidt, Dimitris Tsipras, Aleksander Madry  
Massachusetts Institute of Technology  
{engstrom,ludwigs,tsipras,madry}@mit.edu

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} \delta u \\ \delta v \end{bmatrix}$$

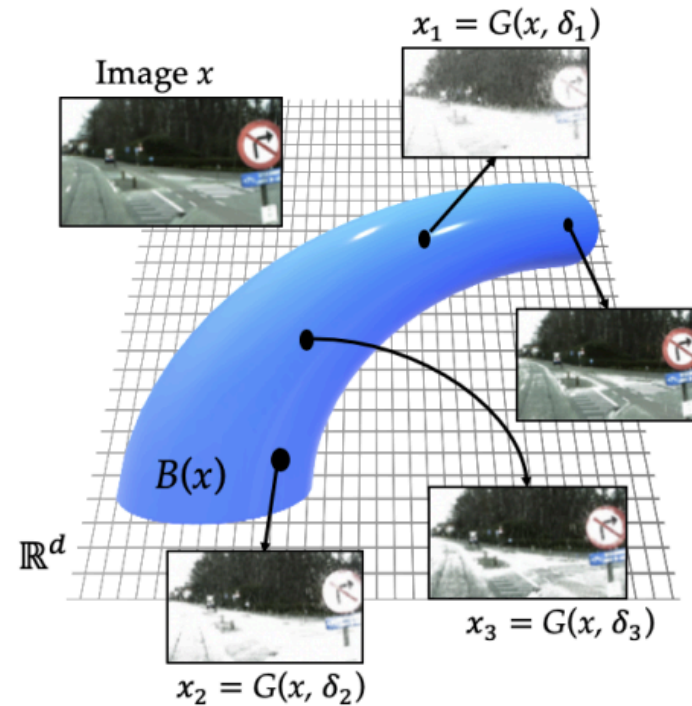
By defining the spatial transformation for some  $x$  as  $T(x; \delta u, \delta v, \theta)$ , we construct an adversarial perturbation for  $x$  by solving the problem

$$\max_{\delta u, \delta v, \theta} \mathcal{L}(x', y), \quad \text{for } x' = T(x; \delta u, \delta v, \theta), \quad (1)$$

# Adversarial Examples Beyond Pixel Perturbations ...



(a) **Perturbation-based robustness.** In perturbation-based adversarial robustness, an adversary can perturb a datum  $x$  into a perceptually similar datum  $x^{\text{adv}} := x + \delta$ . When  $\delta$  is constrained to lie in a set  $\Delta := \{\delta \in \mathbb{R}^d : \|\delta\|_p \leq \epsilon\}$ , the underlying geometry of the problem can be used to find worst-case additive perturbations.

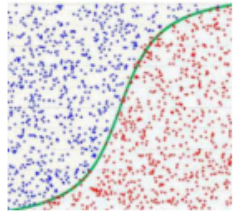


(b) **Model-based robustness.** In our paradigm, models of natural variation can be thought of characterizing a class of *learned image manifolds*  $B(x)$ . By searching over these manifolds, in the model-based robust deep learning paradigm, we seek images  $x' \in B(x)$  that have been subjected to high levels of natural variation.

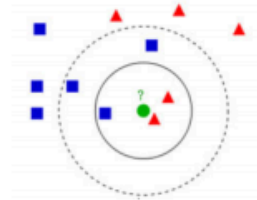


# Adversarial examples...

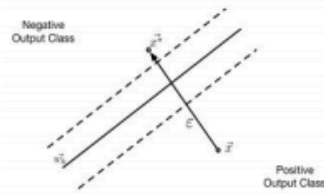
*... beyond deep learning*



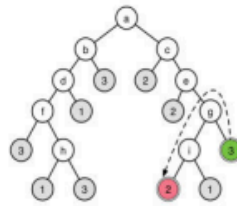
Logistic Regression



Nearest Neighbors

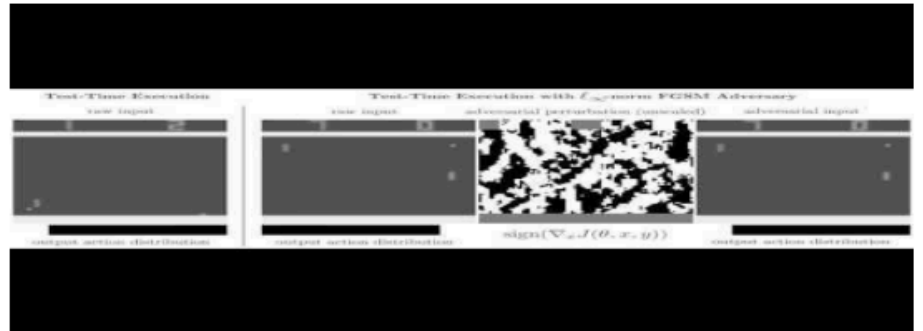
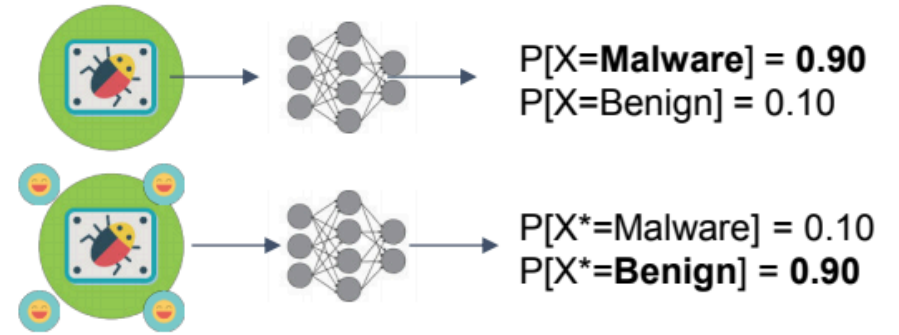


Support Vector Machines



Decision Trees

*... beyond computer vision*



# Poisoning Attack

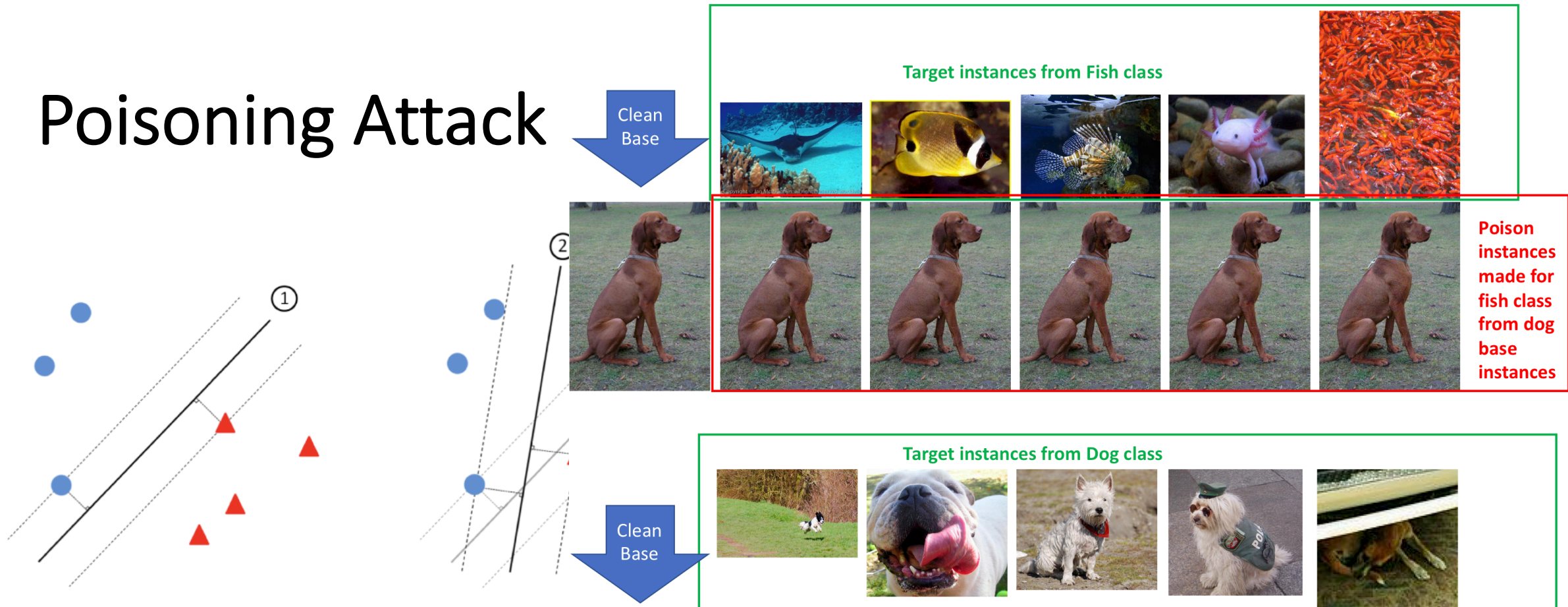


Fig. 1. Linear SVM classifier decision boundary for a two-class dataset with support vectors and class instances. Decision boundary is significantly impacted if just one training sample is changed, even when that sample is a support vector (right).

Let  $f(\mathbf{x})$  denote the function (before the softmax layer). We use  $\mathbf{t}$  as input since it encodes high-level semantic features. Due to the high complexity and nonlinearity of  $f$ , it is possible to find an example  $\mathbf{x}$  that “collides” with the target in feature space, while simultaneously being close to the base instance  $\mathbf{b}$  in input space by computing

$$\mathbf{p} = \underset{\mathbf{x}}{\operatorname{argmin}} \quad \|f(\mathbf{x}) - f(\mathbf{t})\|_2^2 + \beta \|\mathbf{x} - \mathbf{b}\|_2^2 \quad (1)$$

# Model Theft

- **Model theft:** extract model parameters by queries (intellectual property theft)
  - Given a classifier  $F$
  - Query  $F$  on  $q_1, \dots, q_n$  and learn a classifier  $G$
  - $F \approx G$
- *Goals:* leverage active learning literature to develop new attacks and preventive techniques
- *Paper:* *Stealing Machine Learning Models using Prediction APIs*, Tramer et al., Usenix Security 2016
- *Solution:* *Nasty Teacher*, ICLR 2021, et. al.

# Fake News Attacks

## Abusive use of machine learning:

Using GANs to generate **fake content** (a.k.a deep fakes)

*Strong societal implications:*

elections, automated trolling, court

evidence ...



## **Generative media:**

- Video of Obama saying things he never said, ...
- Automated reviews, tweets, comments, indistinguishable from human-generated content

# Towards (Adversarially) Robust ML

- **Algorithms:** Faster robust training + verification [Xiao Tjeng Shafiullah **M** 2018], smaller models, new architectures?
- **Theory:** (Better) adv. robust generalization bounds, new regularization techniques
- **Data:** New datasets and **more comprehensive set of perturbations**

**Major need:** Embracing more of a worst-case mindset

- **Adaptive** evaluation methodology + scaling up verification

**Further Read:** <https://adversarial-ml-tutorial.org/>





The University of Texas at Austin  
**Electrical and Computer  
Engineering**  
*Cockrell School of Engineering*