Visione Artificiale

Raffaella Lanzarotti

Deep Learning
Limitations and New Frontiers

The Rise of Deep Learning

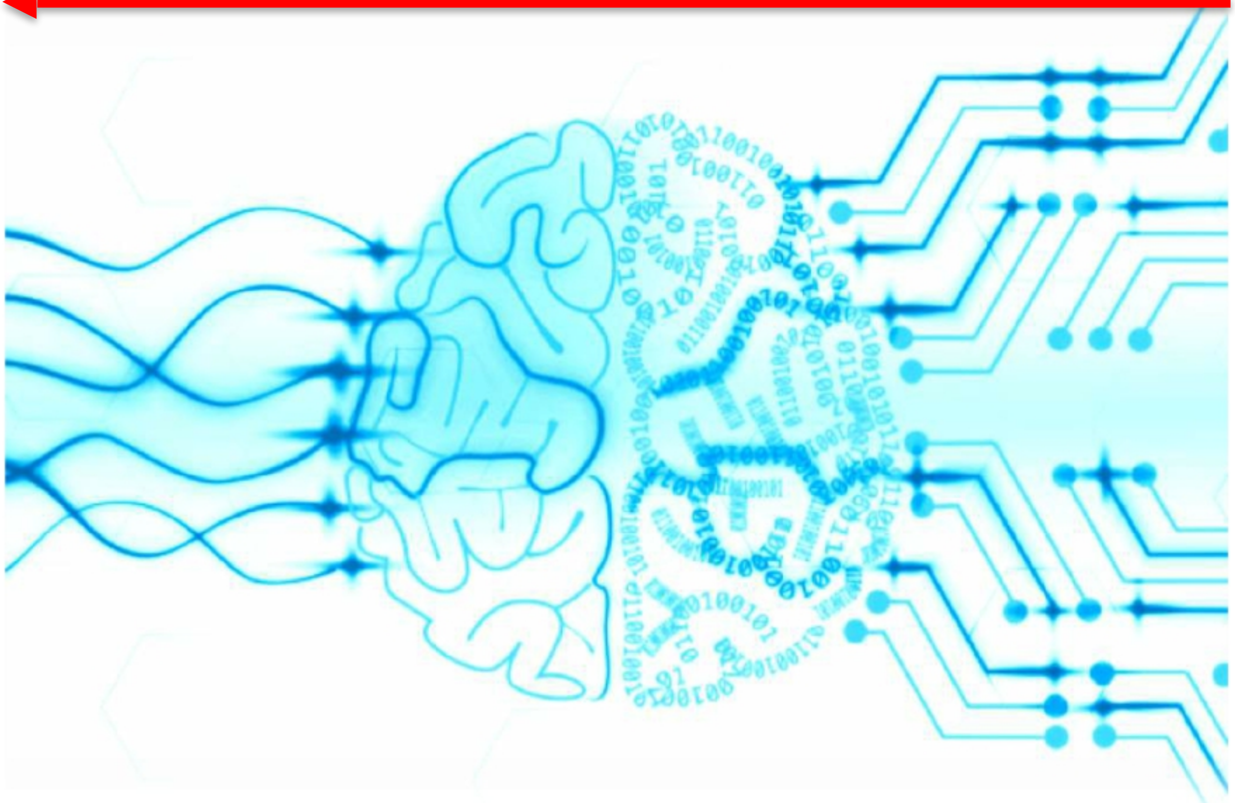
So far..

Generative Models

Data

- Signals
- Images
- Sensors

...



Decision

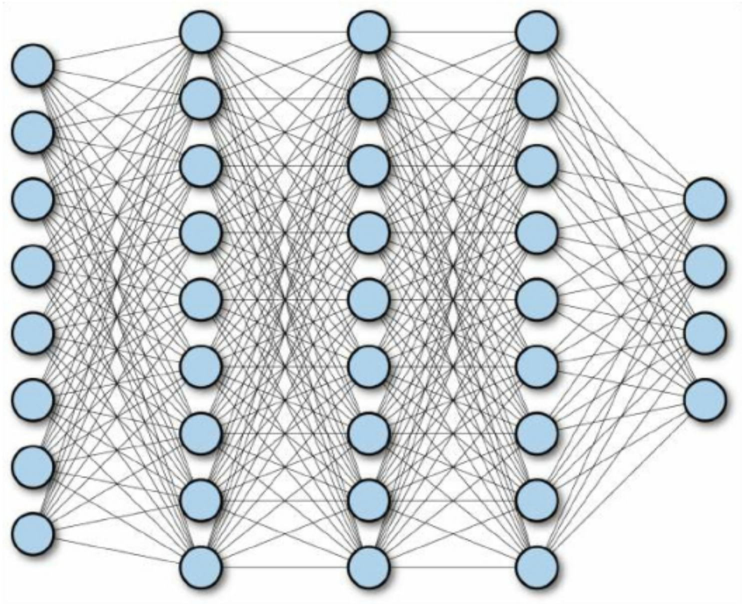
- Prediction
- Detection
- Action

...

Power of Neural Nets

Universal Approximation Theorem

A feedforward network with a single layer is sufficient to approximate, to an arbitrary precision, any continuous function.



Hornik et al. *Neural Networks*. (1989)

Power of Neural Nets

Universal Approximation Theorem

A feedforward network with a single layer is sufficient to approximate, to an arbitrary precision, any continuous function.

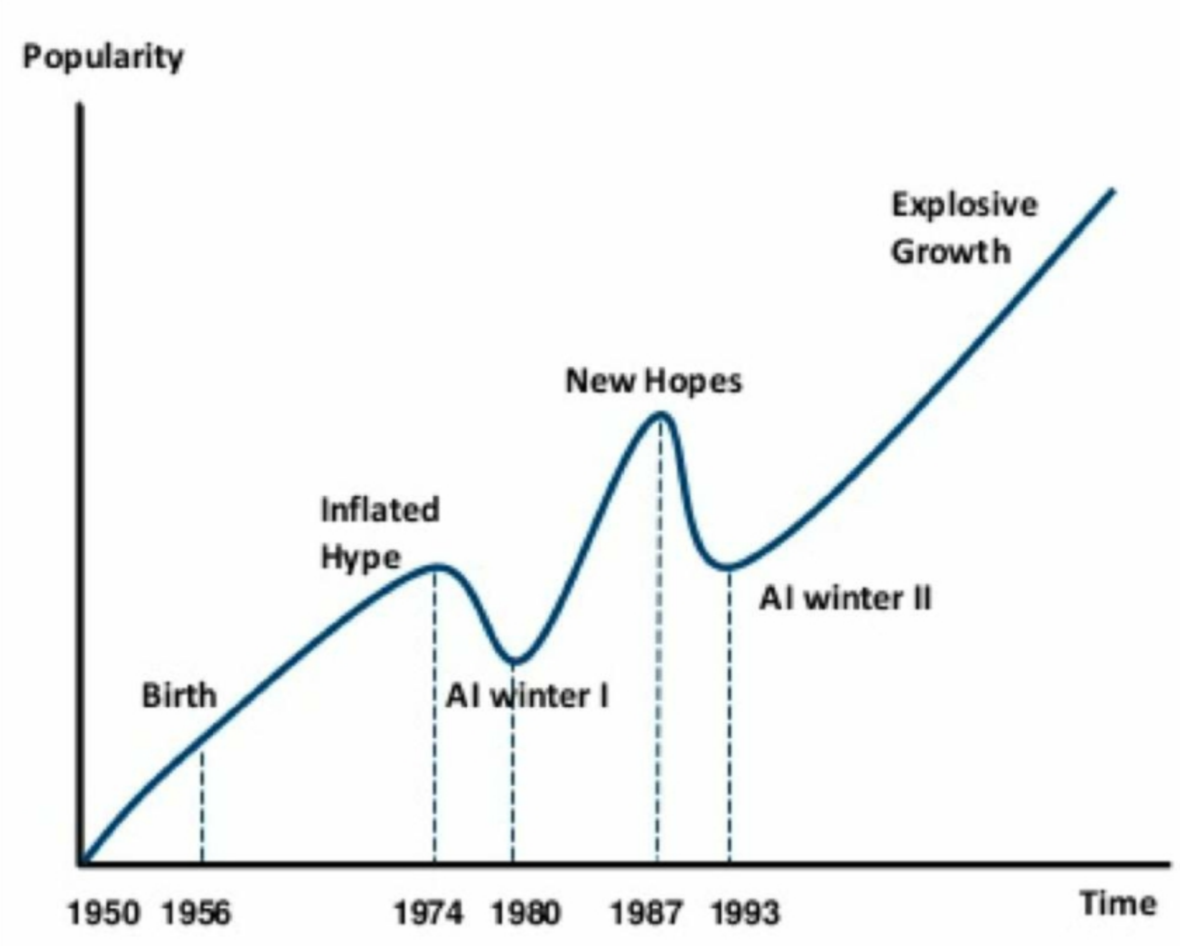
Caveats:

The number of hidden units may be infeasibly large

The resulting model may not generalize

Hornik et al. *Neural Networks*. (1989)

Artificial Intelligence "Hype": Historical Perspective



Limitations

Rethinking Generalization

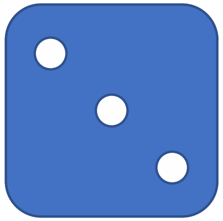
“Understanding Deep Neural Networks Requires Rethinking Generalization

dog

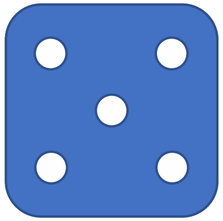
banana

dog

tree

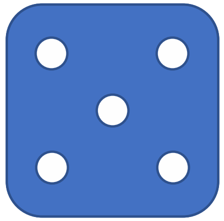


banana



dog

tree

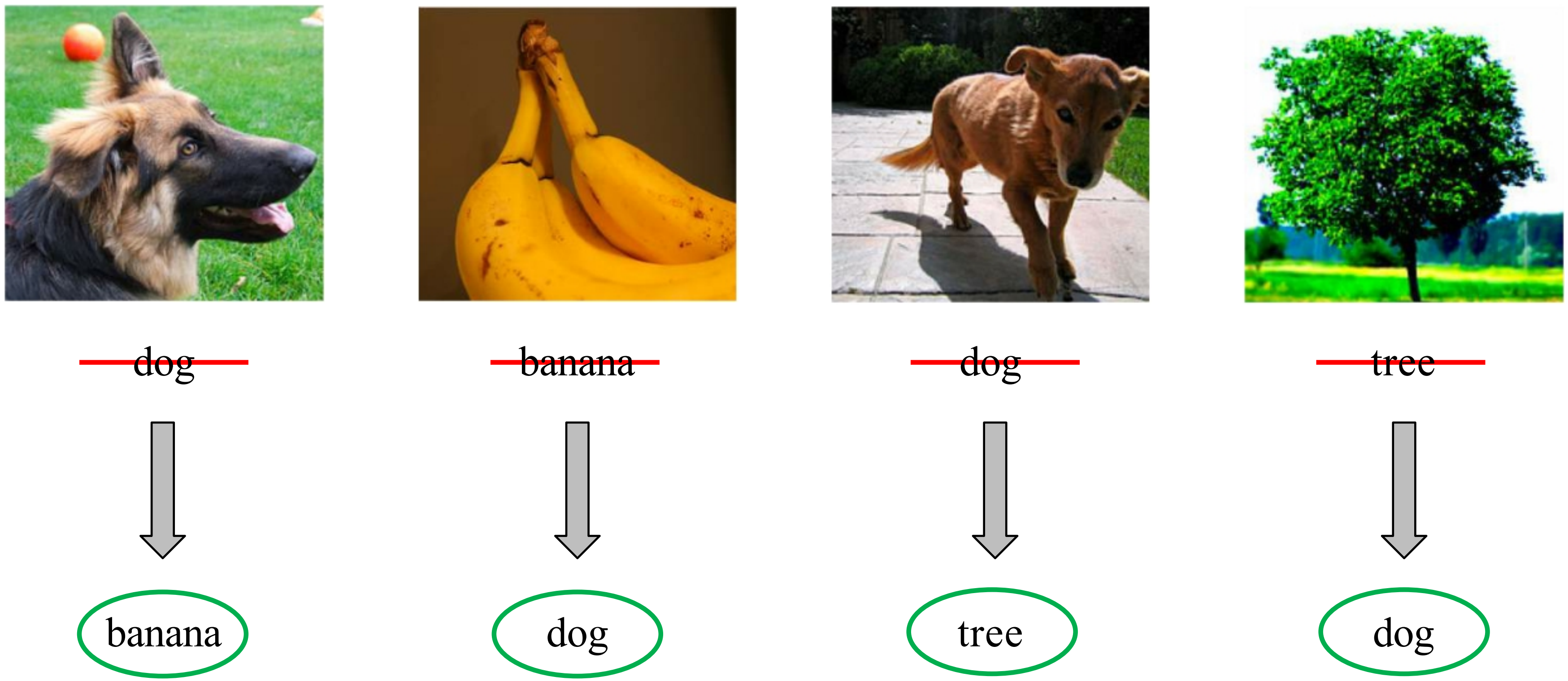


dog

Zhang et al. *ICLR*. (2017)

Rethinking Generalization

"Understanding Deep Neural Networks Requires Rethinking Generalization

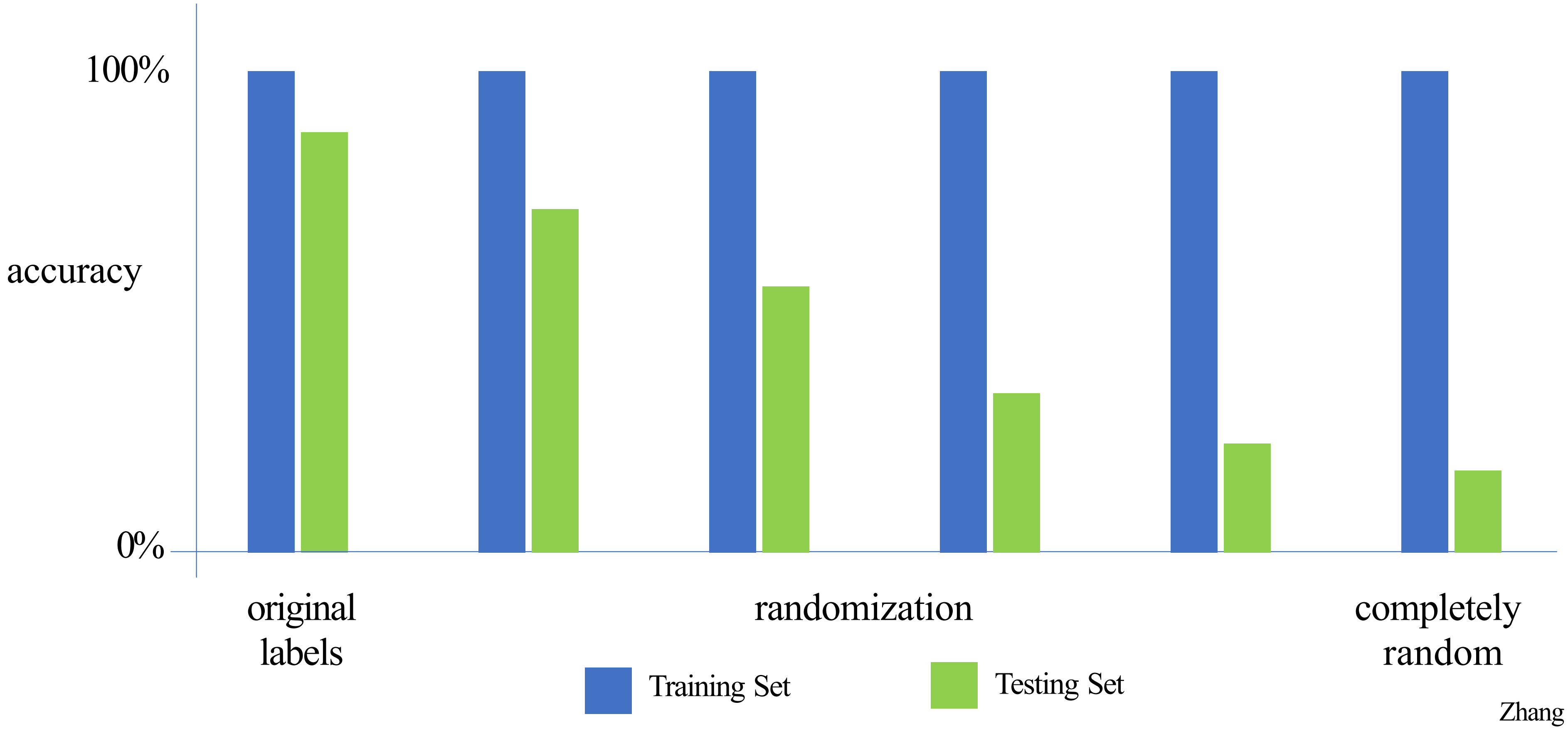


Zhang et al. *ICLR*. (2017)

Capacity of Deep Neural Networks

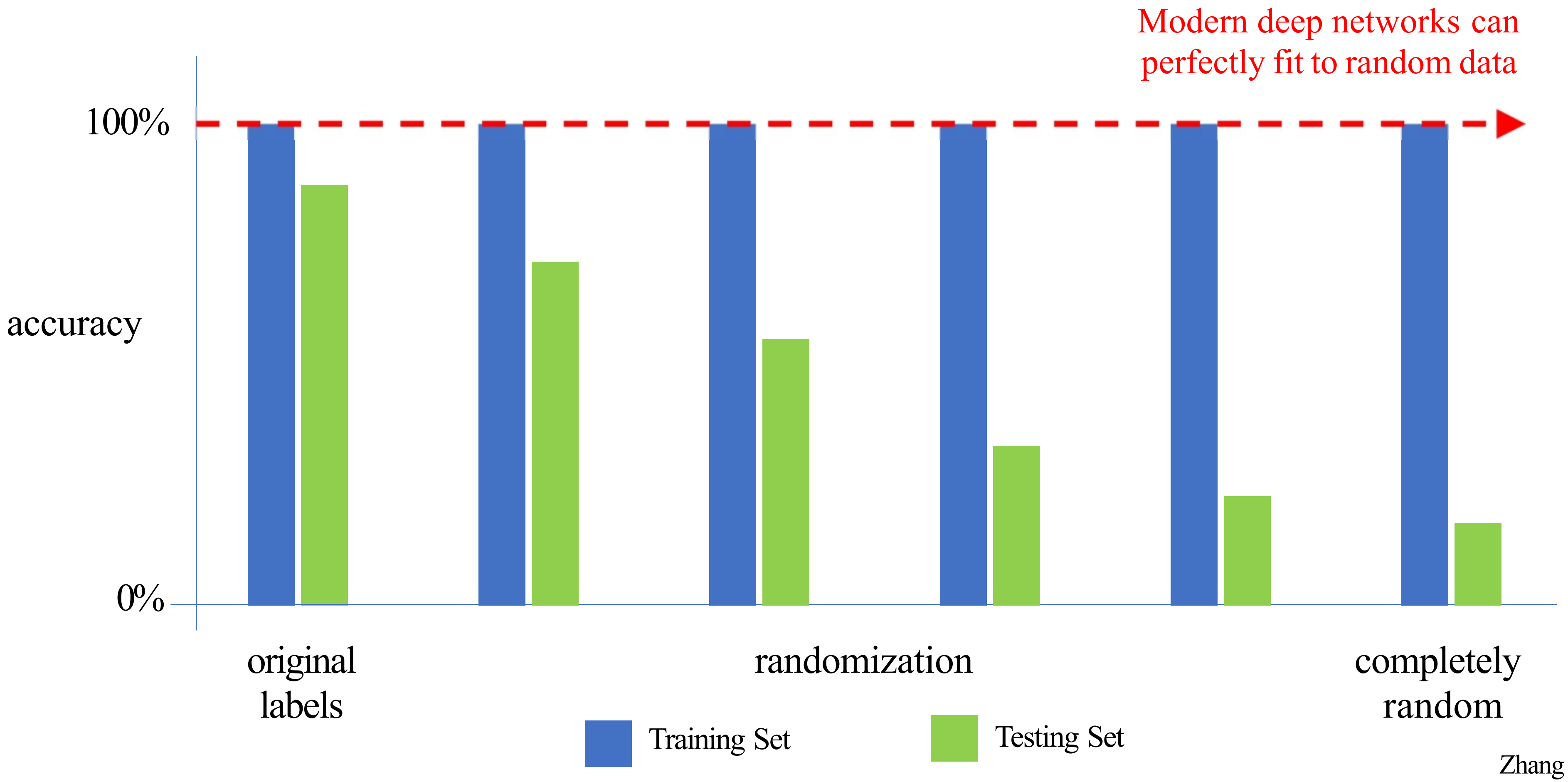
Zhang et al. *ICLR*. (2017)

Capacity of Deep Neural Networks



Zhang et al. *ICLR*. (2017)

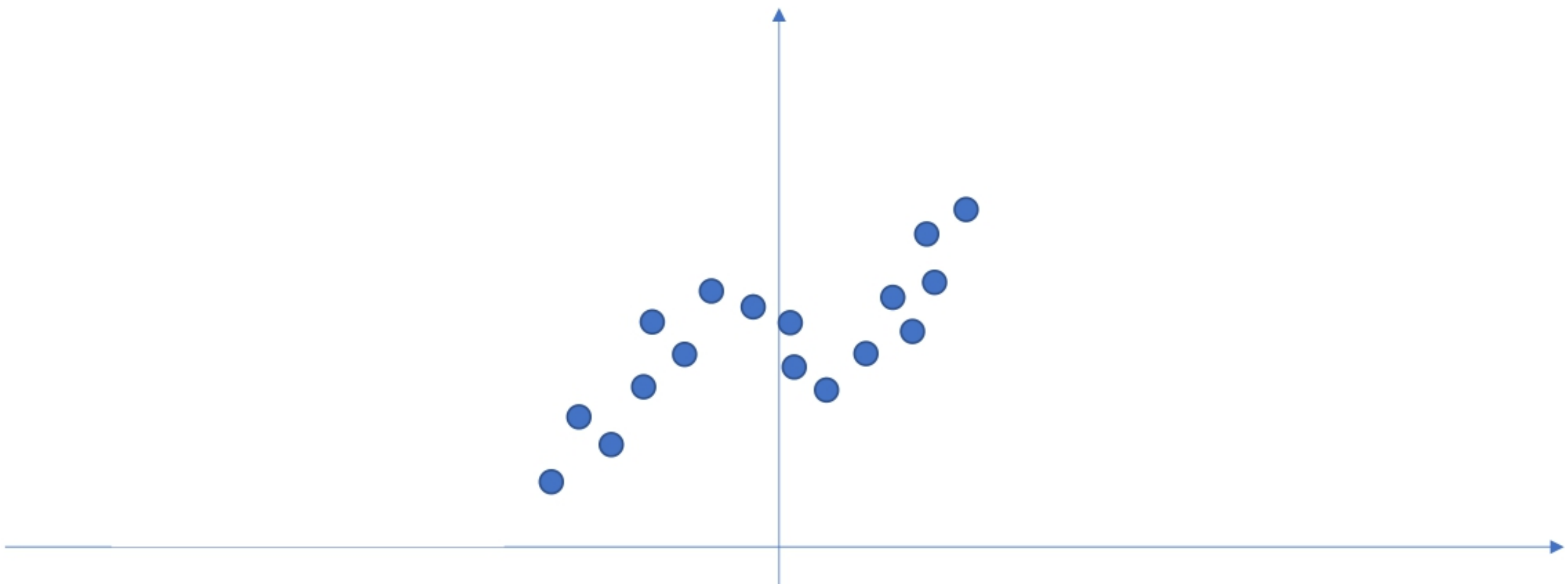
Capacity of Deep Neural Networks



Zhang et al. *ICLR*. (2017)

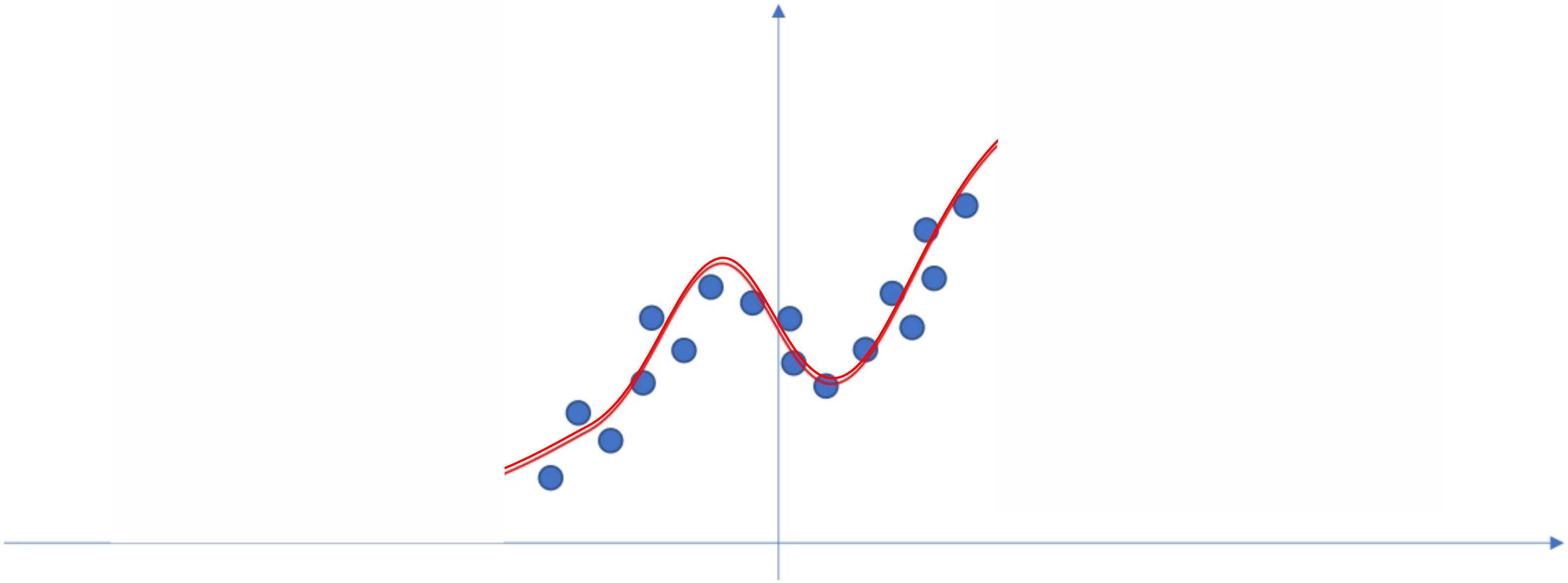
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



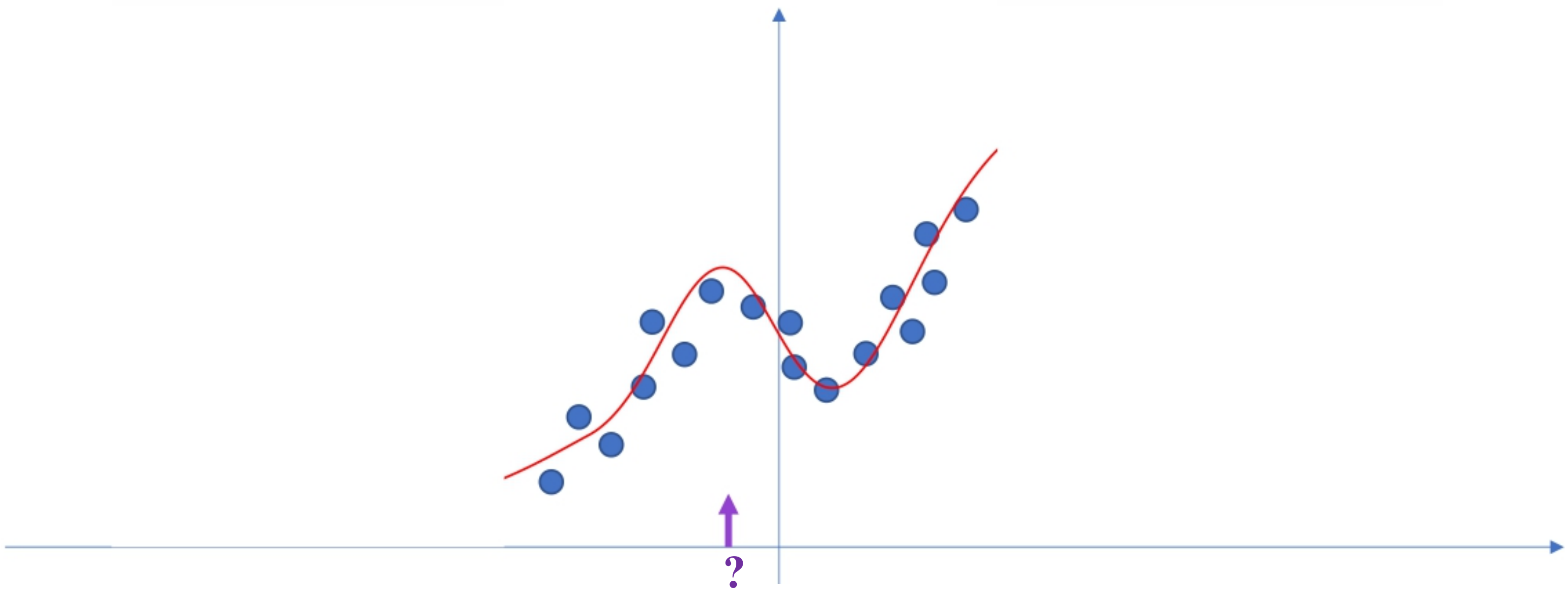
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



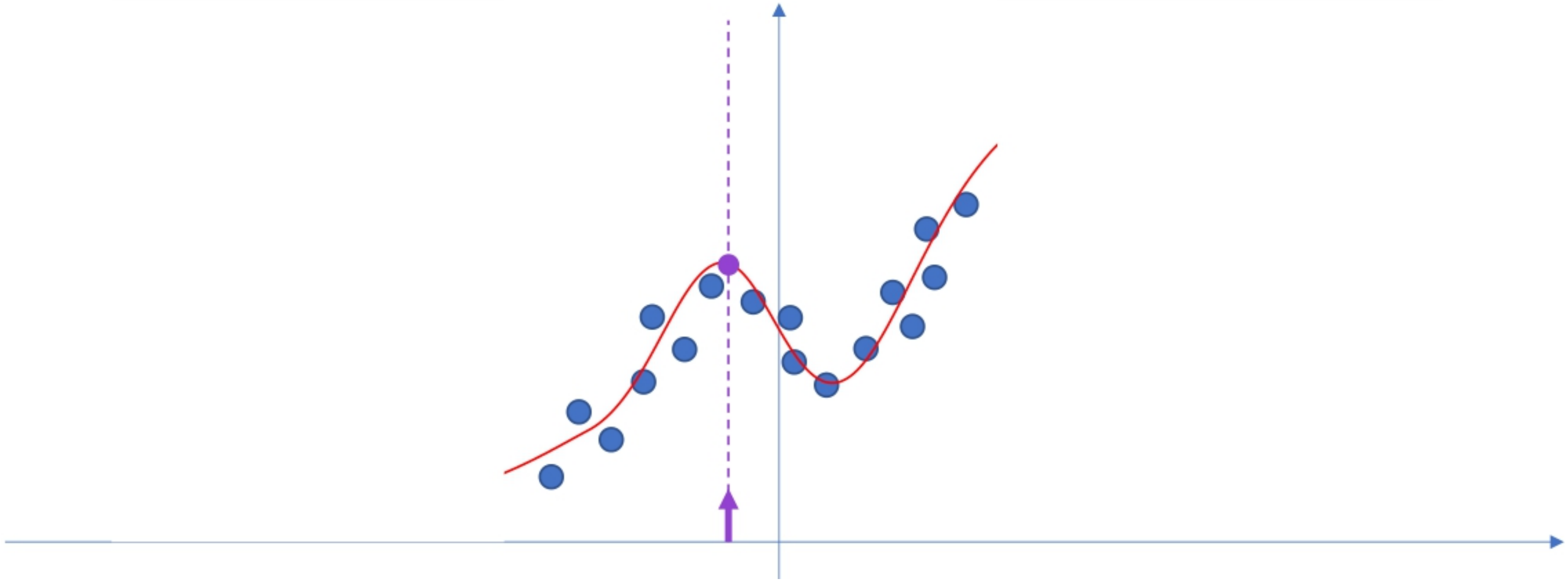
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



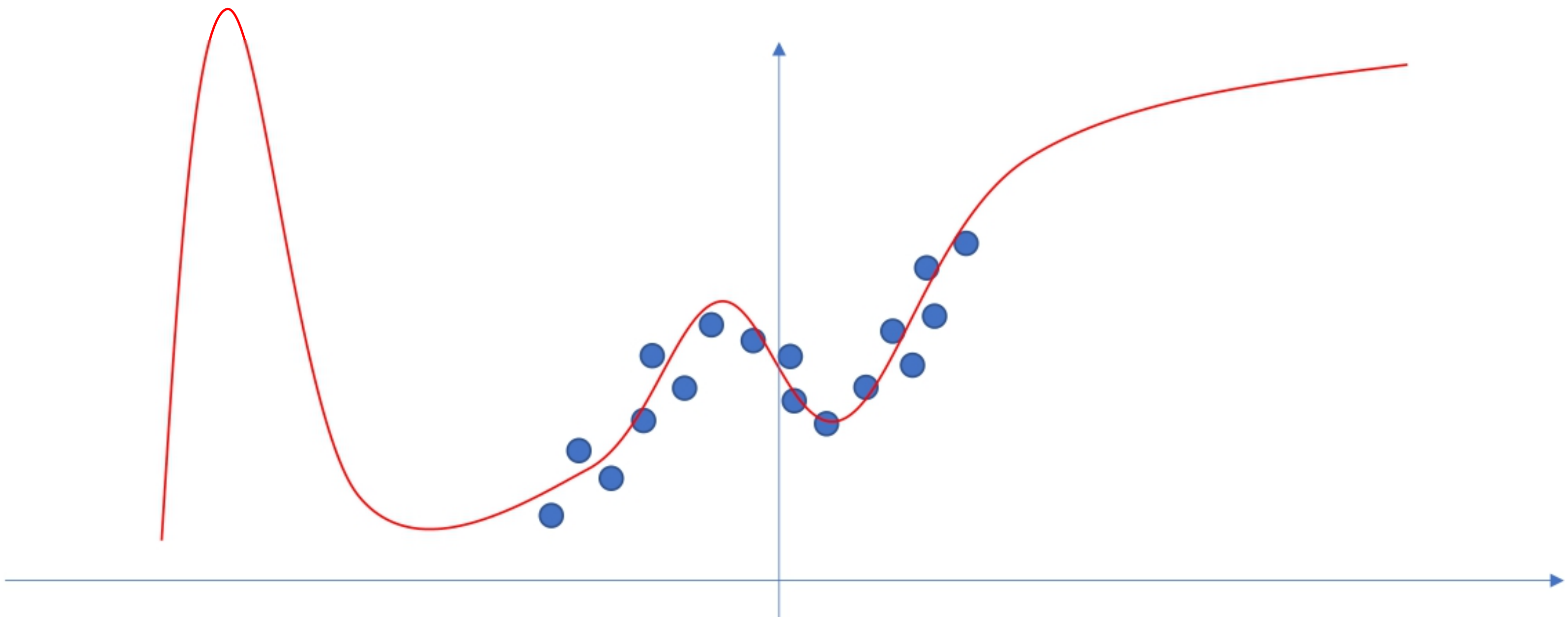
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



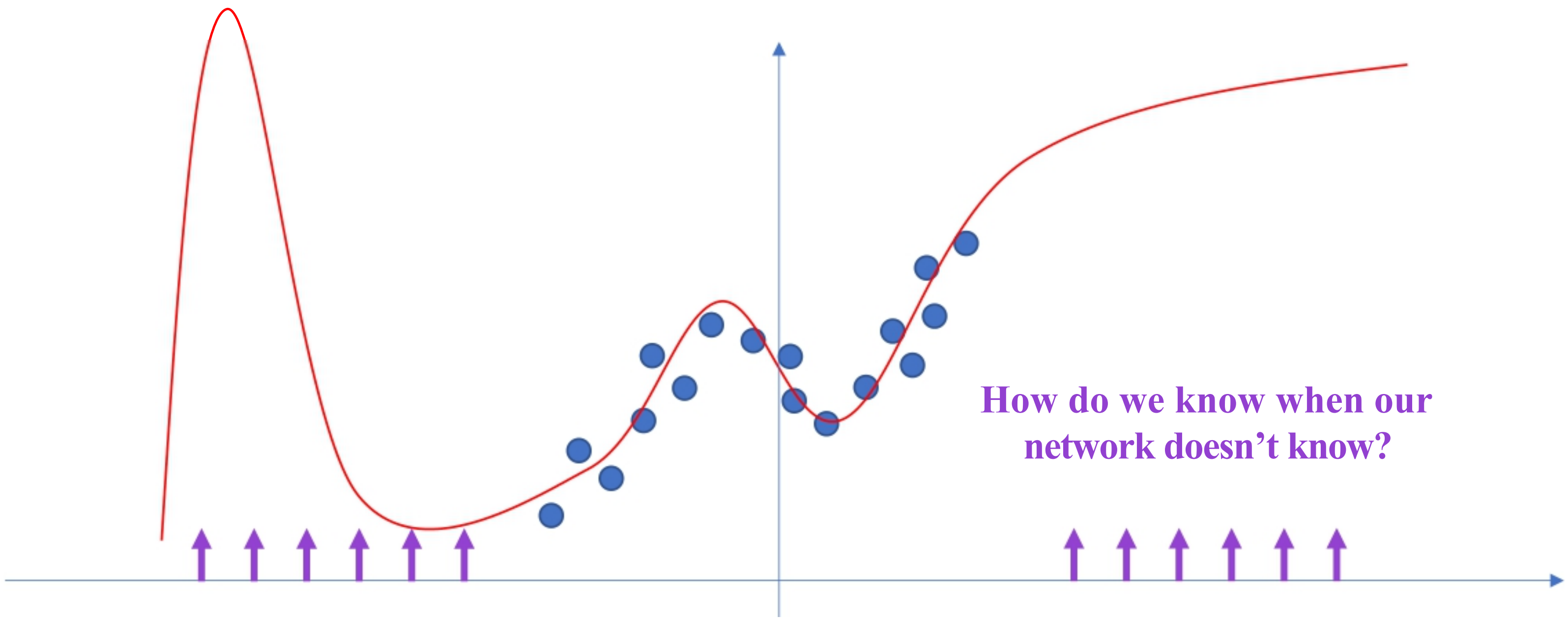
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators

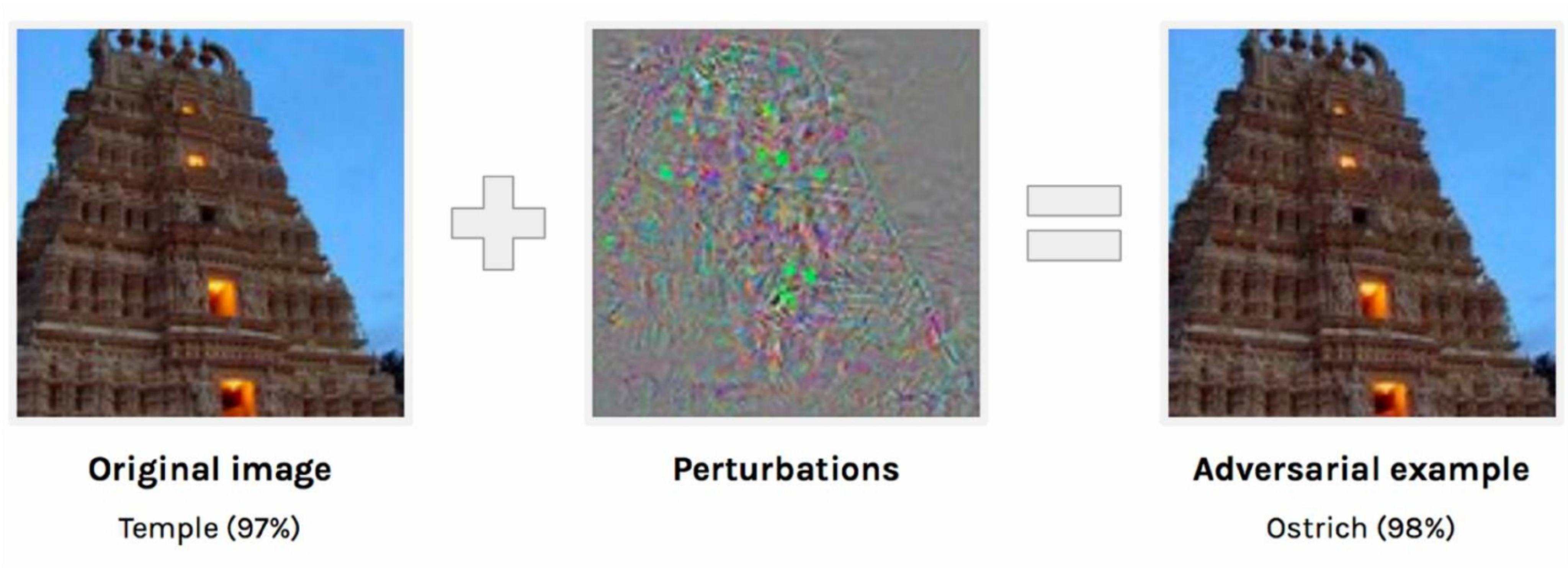


Neural Networks as Function Approximators

Neural networks are **excellent** function approximators
…when they have training data

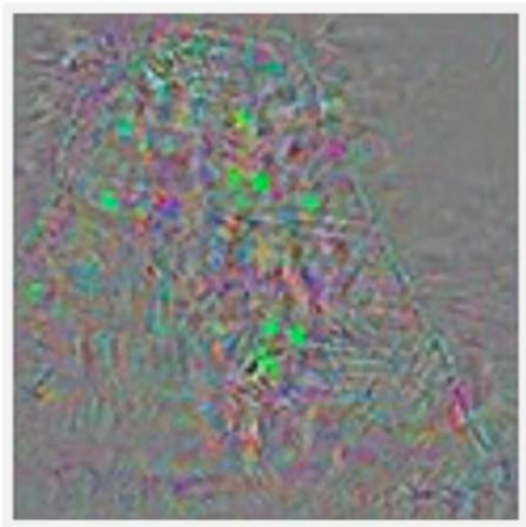


Adversarial Attacks on Neural Networks



Despois. "Adversarial examples and their implications" (2017).

Adversarial Attacks on Neural Networks



Perturbations

Adversarial Attacks on Neural Networks

Remember:

We train our networks with gradient descent

$$\theta \leftarrow \theta - \eta \frac{\partial J(\theta, x, y)}{\partial \theta}$$

"How does a small change in weights decrease our loss"

Adversarial Attacks on Neural Networks

Remember:

We train our networks with gradient descent

$$\theta \leftarrow \theta - \eta \frac{\partial J(\theta, x, y)}{\partial \theta}$$

"How does a small change in weights decrease our loss"

Adversarial Attacks on Neural Networks

Remember:

We train our networks with gradient descent

$$\theta \leftarrow \theta - \eta \frac{\partial J(\theta, x, y)}{\partial \theta}$$

Fix your image x,
and true label y

"How does a small change in weights decrease our loss"

Adversarial Attacks on Neural Networks

Adversarial Image:

Modify image to increase error

$$x \leftarrow x + \eta \frac{\partial J(\theta, x, y)}{\partial x}$$

"How does a small change in the input increase our loss"

Goodfellow et al. *NIPS* (2014)

Adversarial Attacks on Neural Networks

Adversarial Image:

Modify image to increase error

$$\boxed{x} \leftarrow \boxed{x} + \eta \frac{\partial J(\theta, \boxed{x}, y)}{\partial \boxed{x}}$$

"How does a small change in the input increase our loss"

Adversarial Attacks on Neural Networks

Adversarial Image:

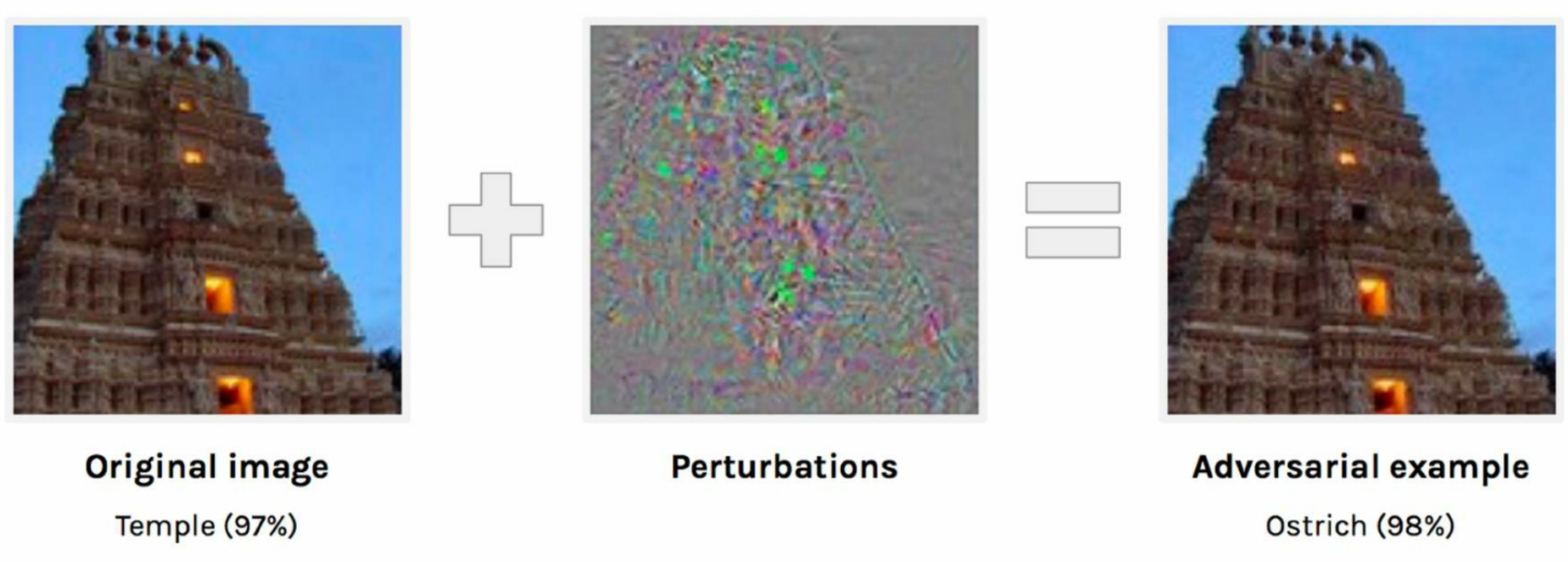
Modify image to increase error

$$x \leftarrow x + \eta \frac{\partial J(\theta, x, y)}{\partial x}$$

Fix your weights $\boldsymbol{\theta}$, and true label $\boldsymbol{y}$

"How does a small change in the input increase our loss"

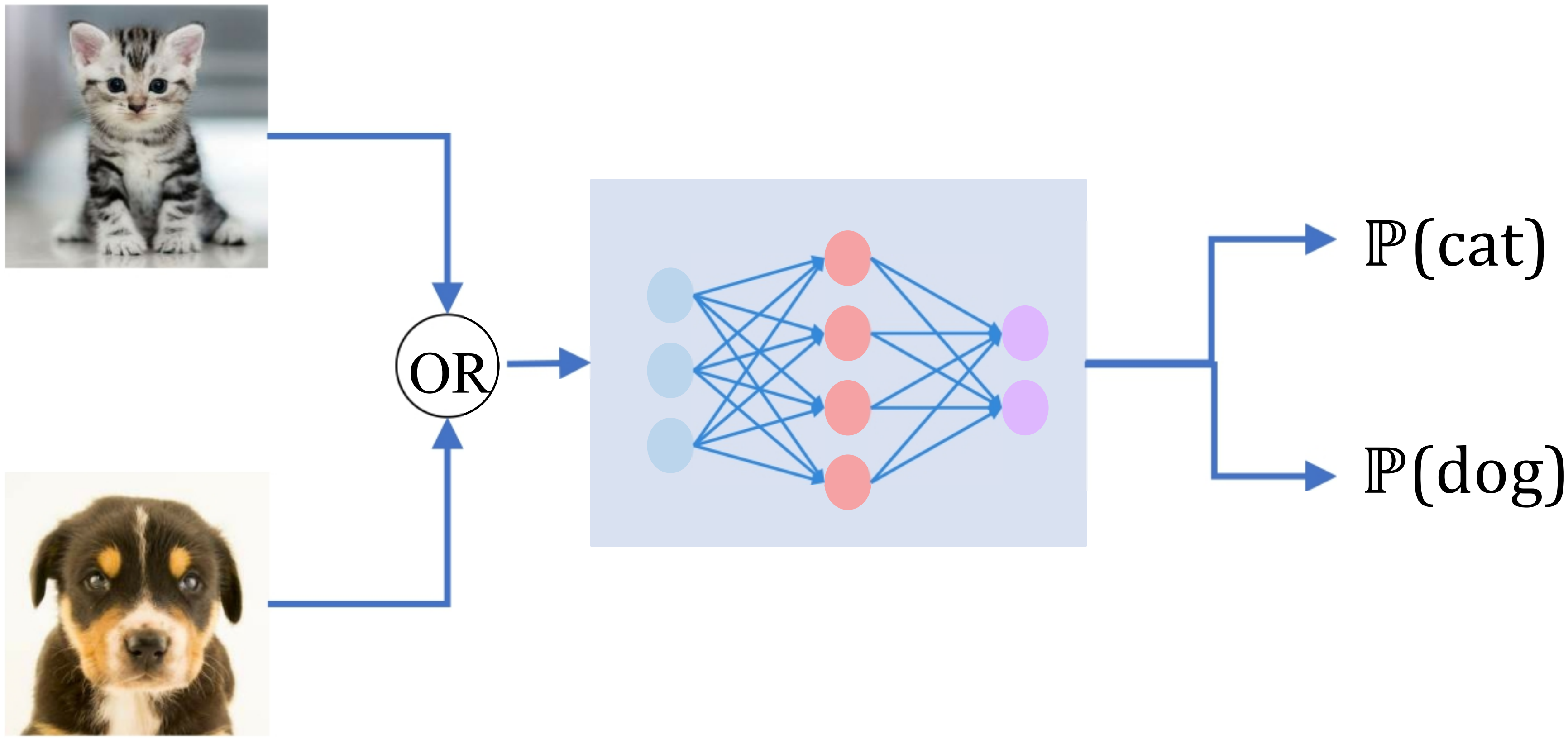
Adversarial Attacks on Neural Networks



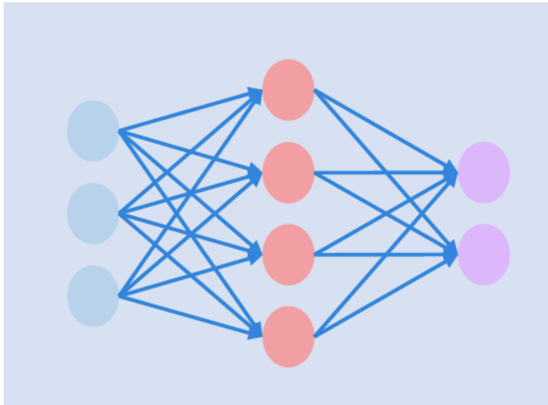
Despois. "Adversarial examples and their implications" (2017).

Bayesian Deep Learning

Why Care About Uncertainty?



Why Care About Uncertainty?



$\mathbb{P}(\text{cat}) = 0.2$

$\mathbb{P}(\text{dog}) = 0.8$

Remember: $\mathbb{P}(\text{cat}) + \mathbb{P}(\text{dog}) = 1$

Bayesian Deep Learning for Uncertainty

Network tries to learn output, $\boldsymbol{Y}$, directly from raw data, $\boldsymbol{X}$

Find mapping, $\boldsymbol{f}$, parameterized by weights $\boldsymbol{\theta}$ such that

$$\min \mathcal{L}(\boldsymbol{Y}, f(X; \boldsymbol{\theta}))$$

Bayesian neural networks aim to learn a posterior over weights, $\mathbb{P}(\boldsymbol{\theta} | \boldsymbol{X}, \boldsymbol{Y})$:

$$\mathbb{P}(\boldsymbol{\theta} | \boldsymbol{X}, \boldsymbol{Y}) = \frac{\mathbb{P}(\boldsymbol{Y} | \boldsymbol{X}, \boldsymbol{\theta}) \mathbb{P}(\boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y} | \boldsymbol{X})}$$

Bayesian Deep Learning for Uncertainty

Network tries to learn output, $\boldsymbol{Y}$, directly from raw data, $\boldsymbol{X}$

Find mapping, $\boldsymbol{f}$, parameterized by weights $\boldsymbol{\theta}$ such that

$$\min \mathcal{L}(\boldsymbol{Y}, f(X; \boldsymbol{\theta}))$$

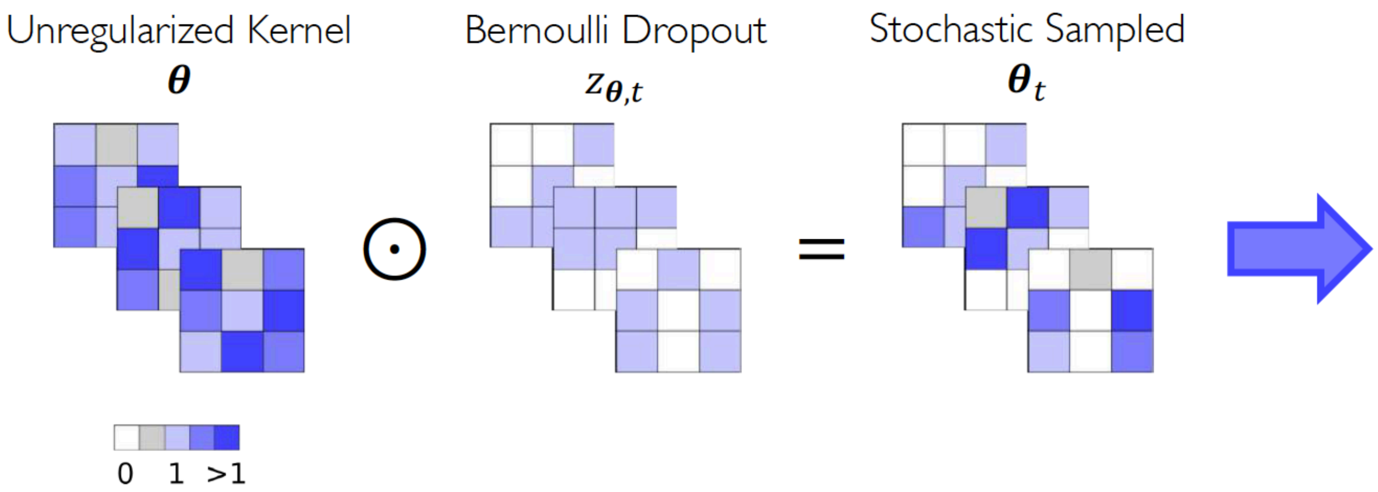
Bayesian neural networks aim to learn a posterior over weights, $\mathbb{P}(\boldsymbol{\theta}|\boldsymbol{X}, \boldsymbol{Y})$:

Intractable! $$\mathbb{P}(\boldsymbol{\theta}|\boldsymbol{X}, \boldsymbol{Y}) = \frac{\mathbb{P}(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{Y}|\boldsymbol{X})}$$

Elementwise Dropout for Uncertainty

Evaluate $\boldsymbol{T}$ stochastic forward passes through the network $\{\boldsymbol{\theta}_t\}_{t=1}^{T}$

Dropout as a form of stochastic sampling $\quad z_{w,t} \sim Bernoulli(p) \quad \forall\, w \in \boldsymbol{\theta}$



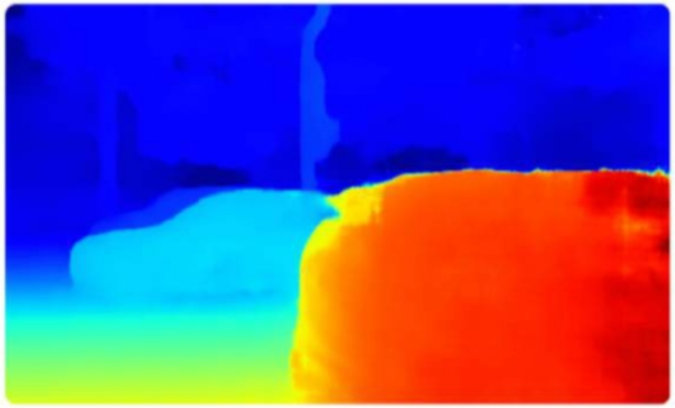
$$\mathbb{E}(\widehat{\boldsymbol{Y}}|\boldsymbol{X}) = \frac{1}{T}\sum_{t=1}^{T} f(\boldsymbol{X}|\boldsymbol{\theta}_t)$$

$$Var(\widehat{\boldsymbol{Y}}|\boldsymbol{X}) = \frac{1}{T}\sum_{t=1}^{T} f(\boldsymbol{X})^2 - \mathbb{E}(\widehat{\boldsymbol{Y}}|\boldsymbol{X})^2$$

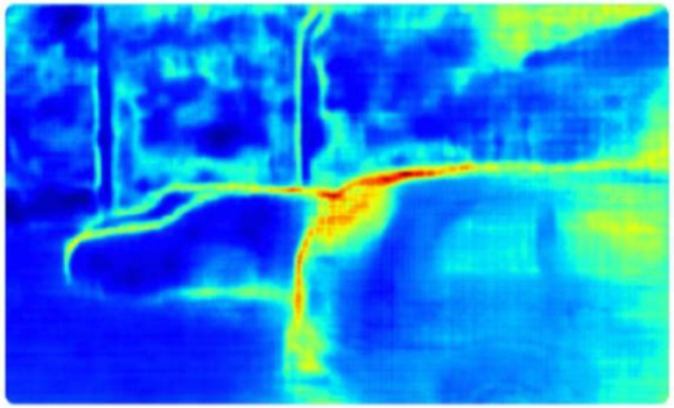
Gal and Ghahramani, *ICML*, 2016.
Amini, Soleimany, et al., *NIPS Workshop on Bayesian Deep Learning*, 2017.

Model Uncertainty Application

Input image



Predicted Depth



Model Uncertainty

Kendall, Gal, *NIPS,* 2017.

Neural Network Limitations…

- Very **data hungry** (eg. often millions of examples)
- **Computationally intensive** to train and deploy (tractably requires GPUs)
- Easily fooled by **adversarial examples**
- Can be subject to **algorithmic bias**
- Poor at **representing uncertainty** (how do you know what the model knows?)
- Uninterpretable **black boxes**, difficult to trust
- **Finicky to optimize**: non-convex, choice of architecture, learning parameters
- Often require **expert knowledge** to design, fine tune architectures

The End