Limitations and New Frontiers

Ava Soleimany

MIT 6.S191

January 29, 2020

T-shirts! Today!

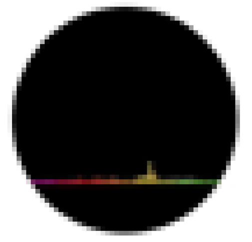
Lecture Schedule

Intro to Deep Learning

Lecture 1
[Slides] [Video] *coming soon!*

Deep Sequence Modeling

Lecture 2
[Slides] [Video] *coming soon!*



Intro to Tensorflow; Music Generation

Lab Session 1
[Code] *coming soon!*

Deep Computer Vision

Lecture 3
[Slides] [Video] *coming soon!*

Deep Generative Modeling

Lecture 4
[Slides] [Video] *coming soon!*

De-biasing Facial Recognition Systems

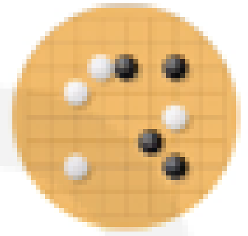
Lab Session 2
[Code] [Paper] *coming soon!*

Deep Reinforcement Learning

Lecture 5
[Slides] [Video] *coming soon!*

Limitations and New Frontiers

Lecture 6
[Slides] [Video] *coming soon!*



Pixels-to-Control Learning

Lab Session 3
[Code] *coming soon!*

Guest Lecture

Lecture 7
[Info] [Slides] [Video] *coming soon!*

Robot Learning

Lecture 8
[Info] [Slides] [Video] *coming soon!*

Final Projects

Lab Session 4
[Video] *coming soon!*

Neural Rendering

Lecture 9
[Info] [Slides] [Video] *coming soon!*

ML for Scent

Lecture 10
[Info] [Slides] [Video] *coming soon!*

Final Projects and Awards Ceremony

Lab Session 5
[Video] *coming soon!*

- Mon Jan 27 – Fri Jan 31
- 1:00 pm – 4:00pm, 32-123
- Lecture + Lab Breakdown
- Graded P/D/F; 3 Units
- 1 Final Assignment
- Lab submissions: Thursday 1/30, 5pm

Final Class Project

Option 1: Proposal Presentation

- At least 1 registered student to be prize eligible
- Present a novel deep learning research idea or application
- 3 minutes (strict)
- Presentations on **Friday, Jan 31**
- Submit groups by **Wednesday 11:59pm** to be eligible
- Submit slide by **Thursday 11:59pm** to be eligible
- Instructions: **shorturl.at/wxBK7**

- Judged by a panel of judges
- Top winners are awarded:

3x NVIDIA 2080 Ti ($4000)

4x Google Home ($400)

3x Display Monitors ($300)

3x SSD 1TB ($200)

Final Class Project

Option 1: Proposal Presentation

- At least 1+ registered student to be prize eligible
- Present a novel deep learning research idea or application
- 3 minutes (strict)
- Presentations on **Friday, Jan 31**
- Submit groups by **Wednesday 11:59pm** to be eligible
- Submit slide by **Thursday 11:59pm** to be eligible
- Instructions: **shorturl.at/wxBK7**

Proposal Logistics

- Prepare slides on Google Slides
- **Group submit by today 11:59pm:** **shorturl.at/mxBWZ**
- In class project work: **Thu, Jan 30**
- **Slide submit by Thu 11:59 pm:** **shorturl.at/pqCL9**
- Presentations on **Friday, Jan 31**

Final Class Project

Option 1: Proposal Presentation

- At least 1+ registered student to be prize eligible
- Present a novel deep learning research idea or application
- 3 minutes (strict)
- Presentations on **Friday, Jan 31**
- Submit groups by **Wednesday 11:59pm** to be eligible
- Submit slide by **Thursday 11:59pm** to be eligible
- Instructions: shorturl.at/wxBK7

Option 2: Write a 1-page review of a deep learning paper

- Grade is based on clarity of writing and technical communication of main ideas
- Due **Friday Jan 31 1:00pm** (before lecture) by email

Thursday: AI for Human Creativity + Robot Learning

David Cox,
IBM Director,
MIT-IBM Watson AI Lab
Towards Robust AI

IBM **Research**

Animesh Garg,
U Toronto,
NVIDIA
Robot Learning

NVIDIA.

Lab + Final Project Work

Ask us questions!

Open office hours!

Work with group members!

Friday: Neural Rendering + Learning to Smell
Project Proposals + Awards!

Chuan Li,
CSO,
Lambda Labs
Neural Rendering

Lambda

Alex Wiltschko,
Senior Research Scientist,
Google Brain
Machine Learning for Scent

Google

Project Proposals!

Judging and Awards!

Pizza Celebration!

So far in 6.S191…

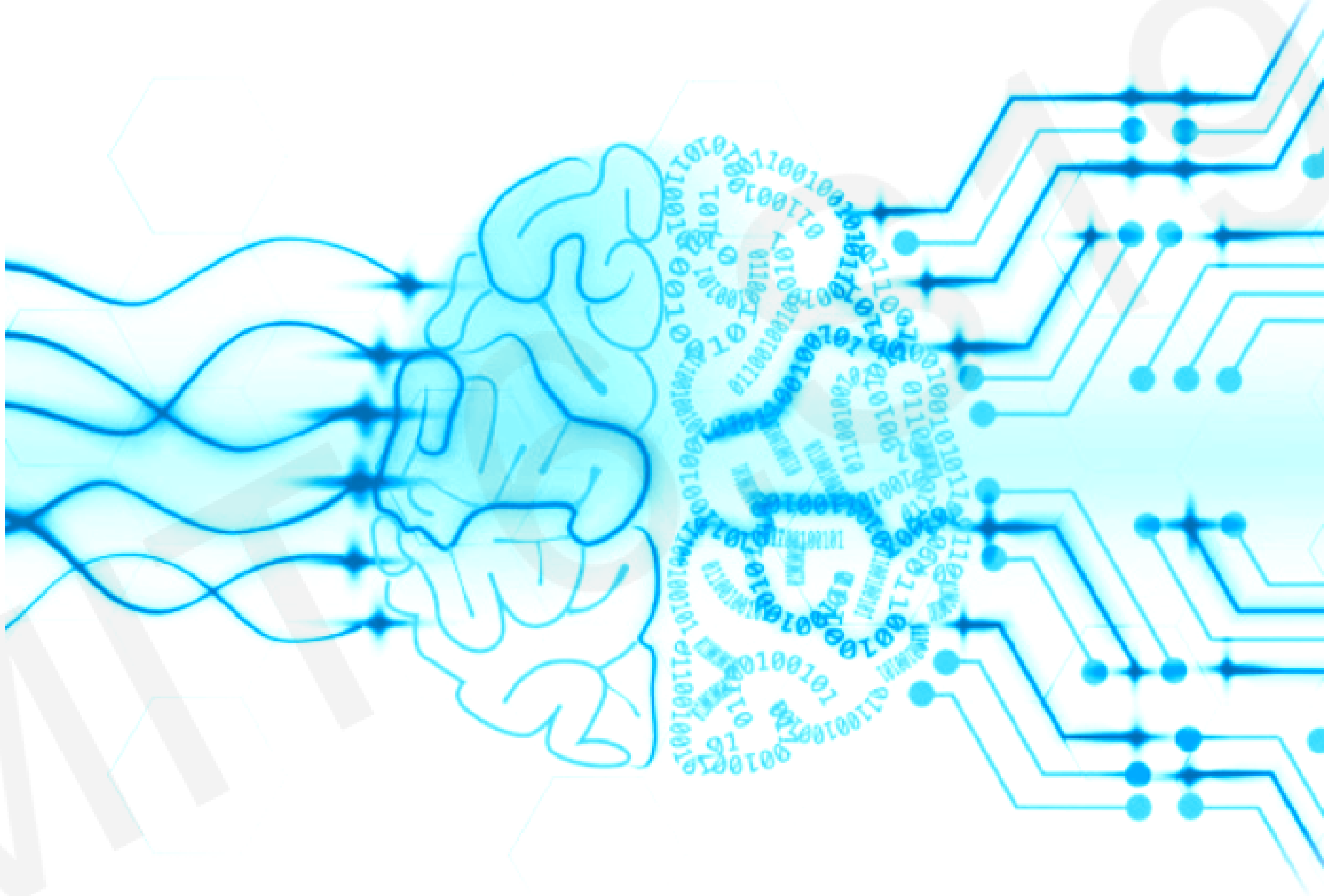
The Rise of Deep Learning

So far in 6.S191…

Data

- Signals
- Images
- Sensors

…

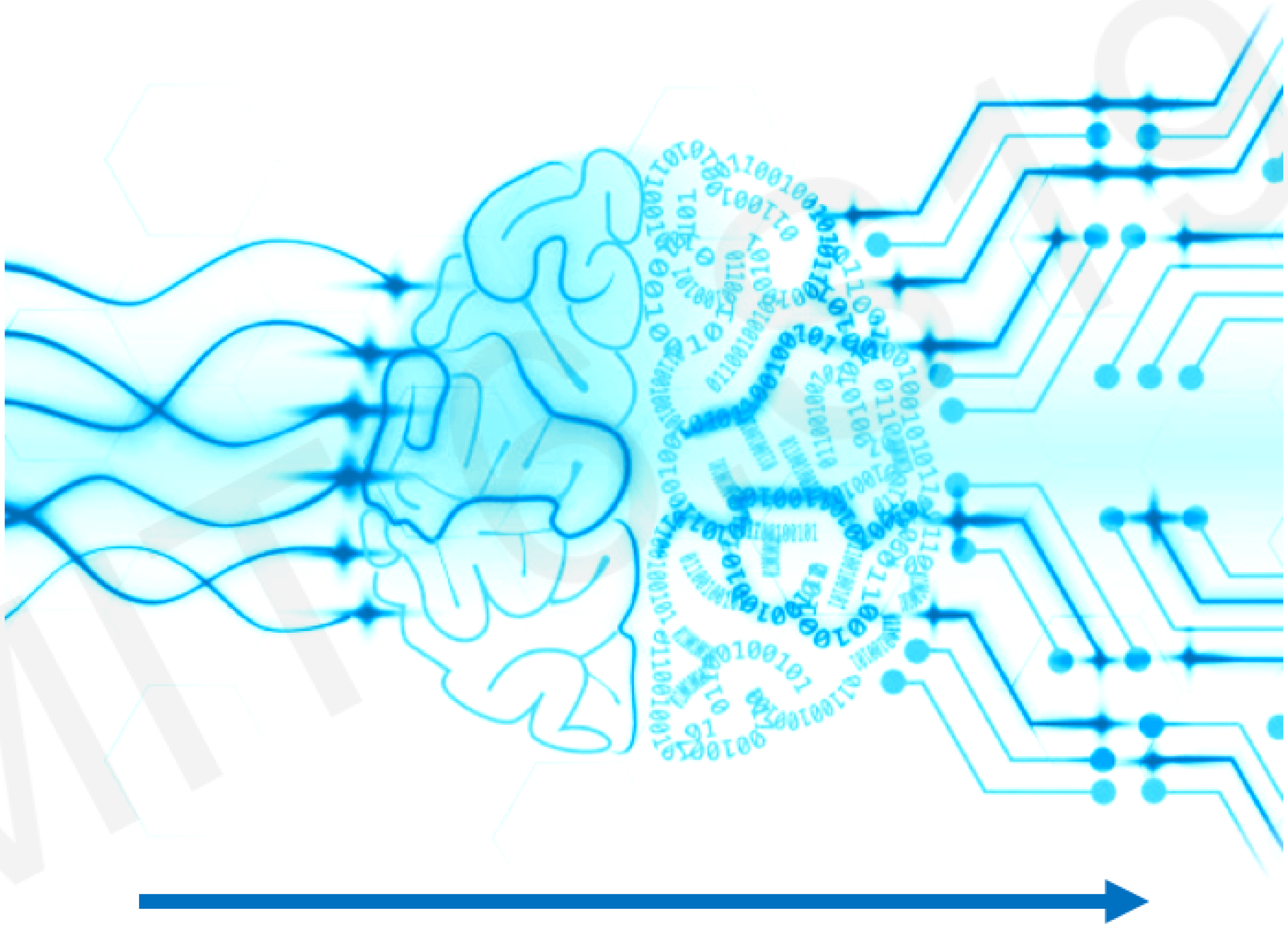


So far in 6.S191…

Data

- Signals
- Images
- Sensors

…

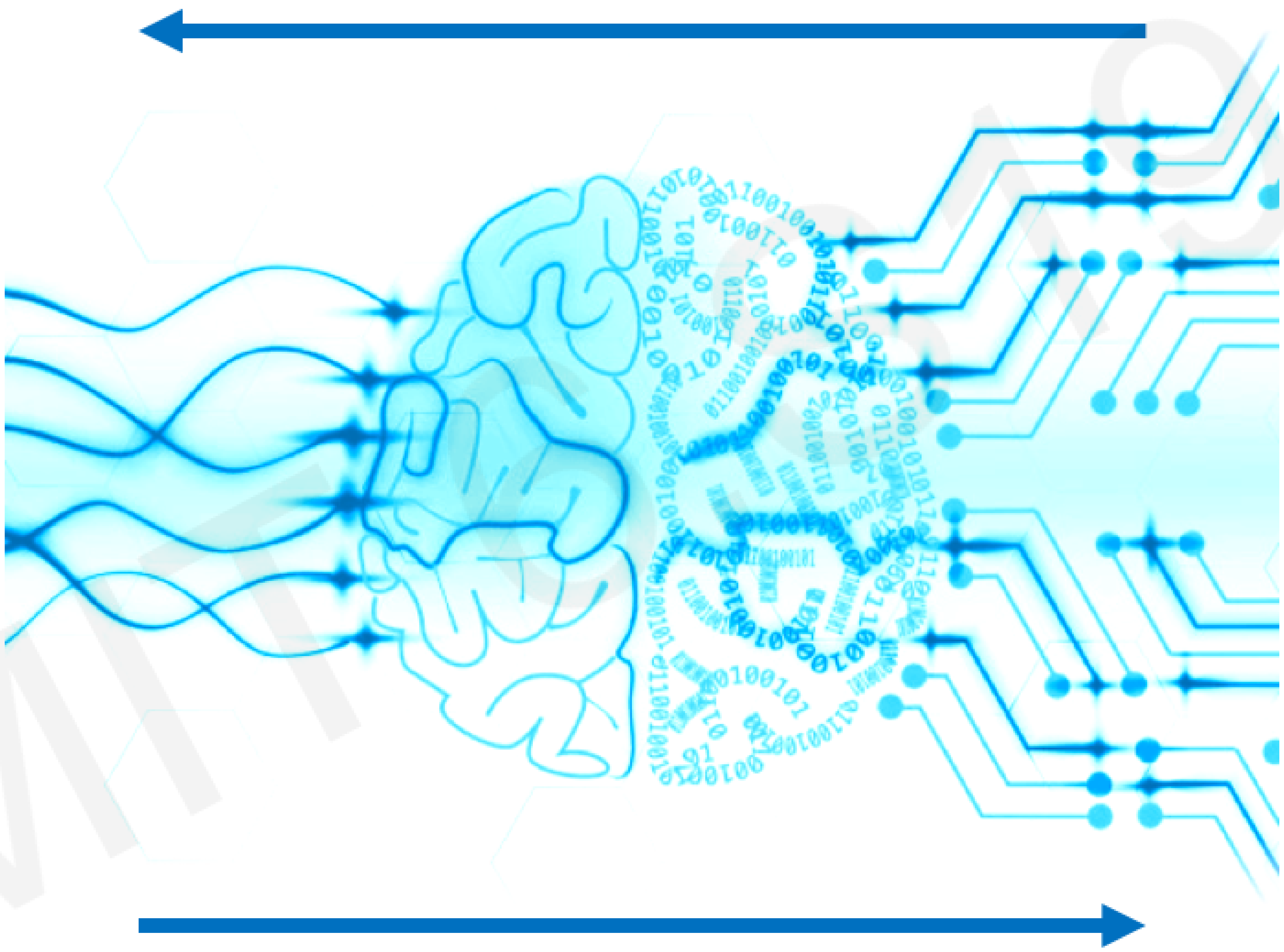


Decision

- Prediction
- Detection
- Action

…

So far in 6.S191…



Data
- Signals
- Images
- Sensors

…

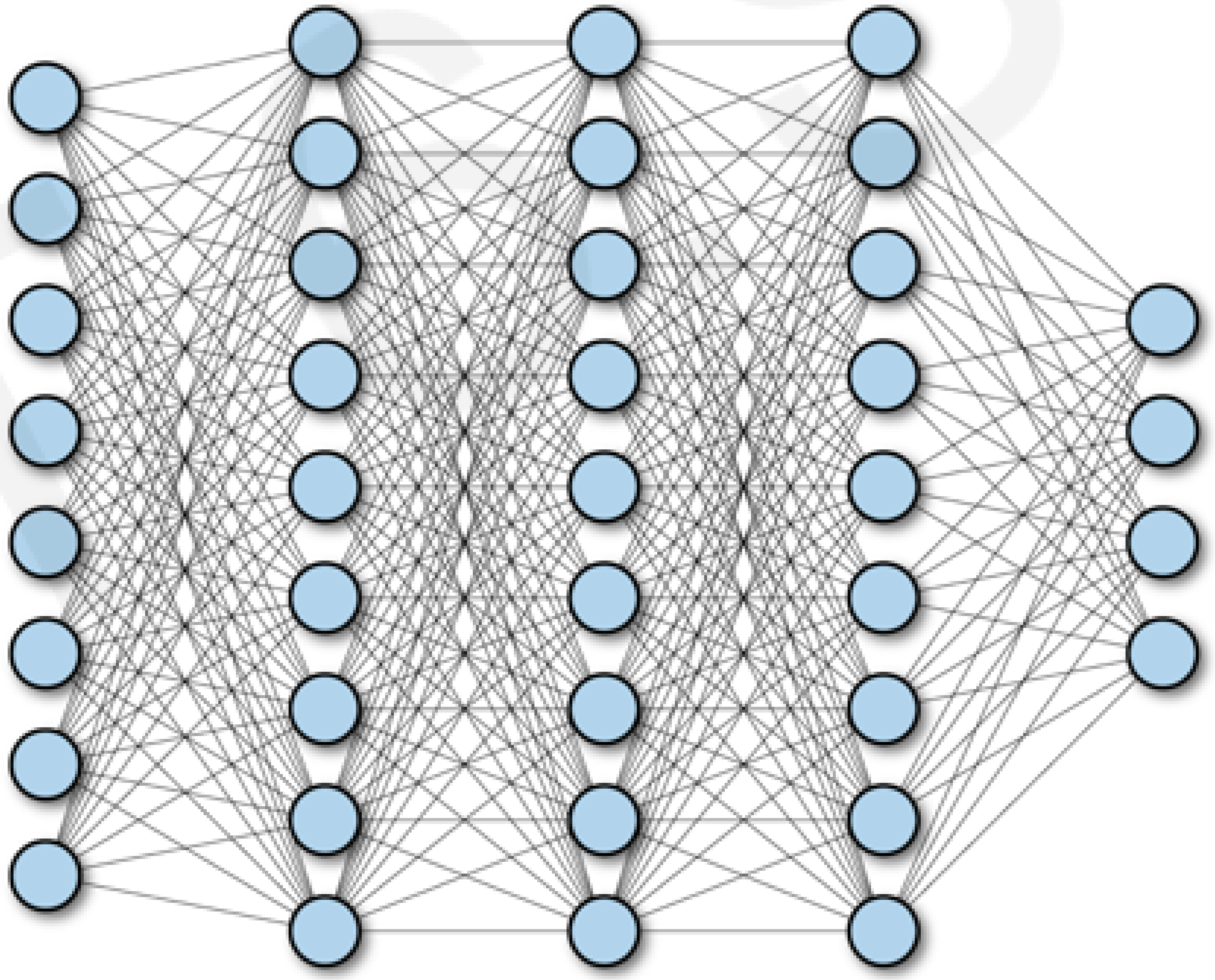
Decision
- Prediction
- Detection
- Action

…

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.



Power of Neural Nets

Universal Approximation Theorem

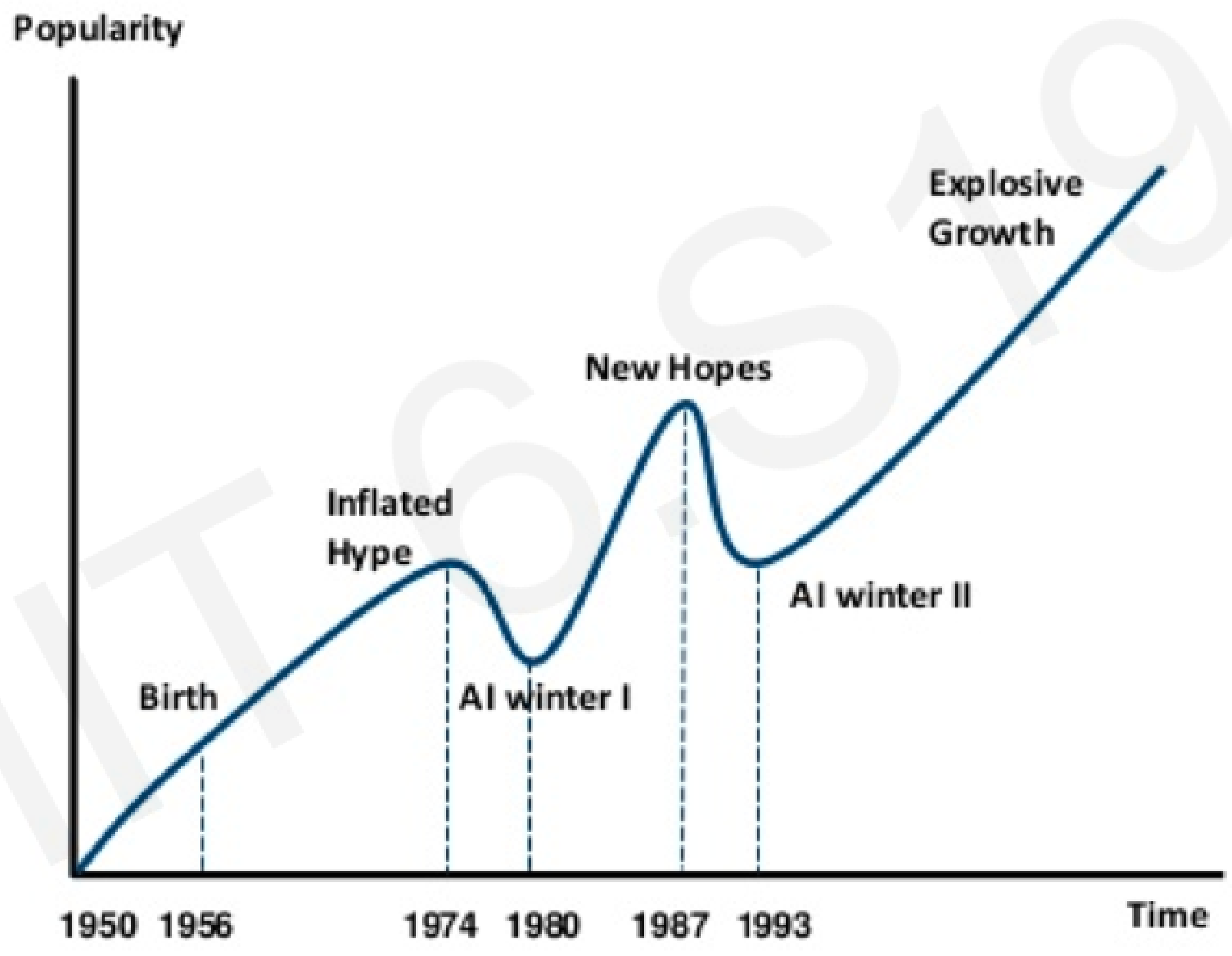
A feedforward network with a single layer is sufficient to approximate, to an arbitrary precision, any continuous function.

Caveats:

The number of hidden units may be infeasibly large

The resulting model may not generalize

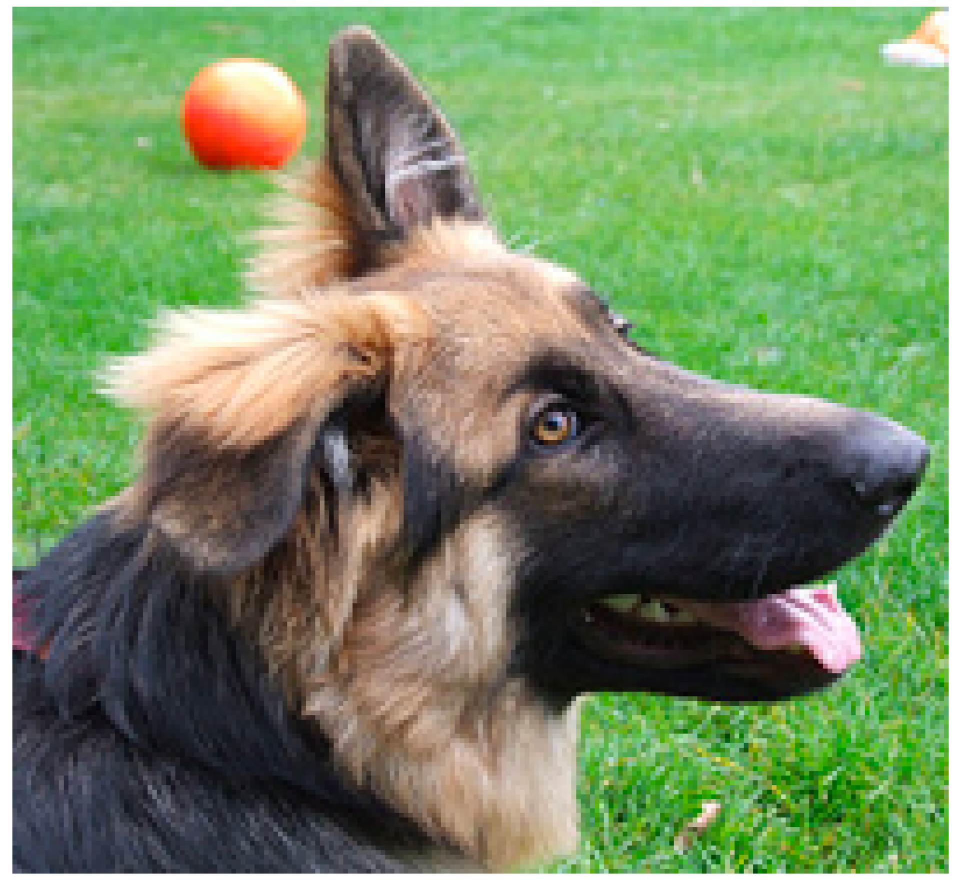
Artificial Intelligence “Hype”: Historical Perspective



Limitations

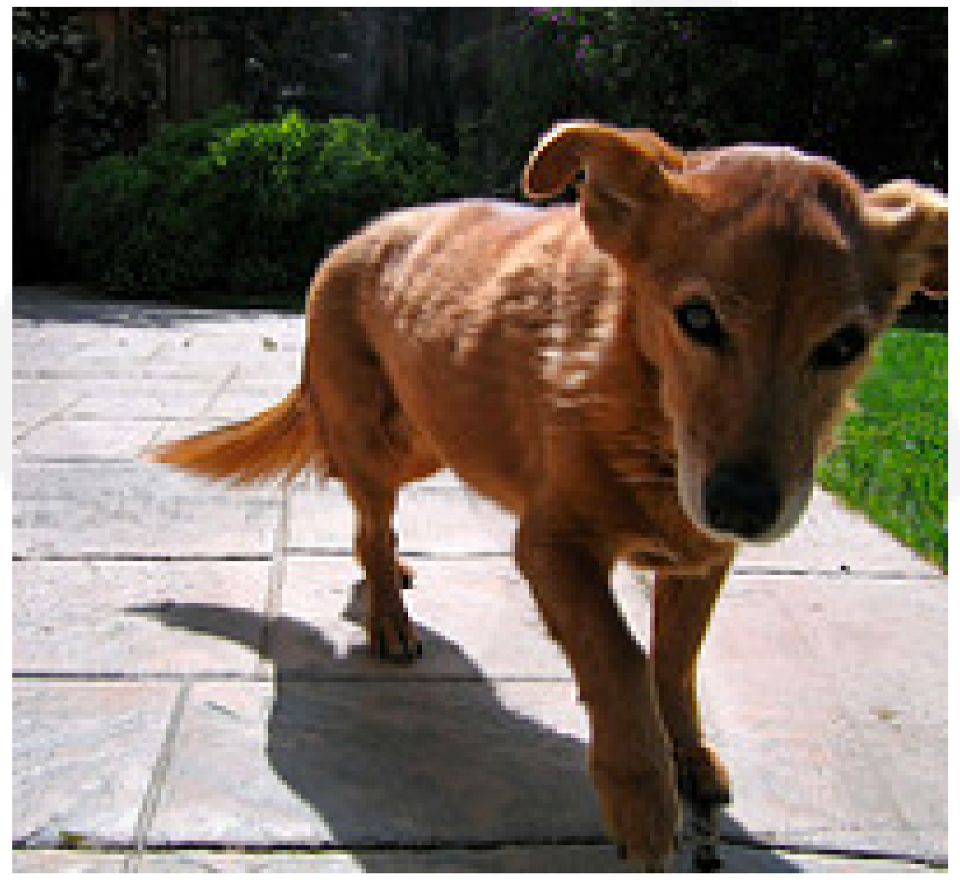
Rethinking Generalization

"Understanding Deep Neural Networks Requires Rethinking Generalization"



dog

banana

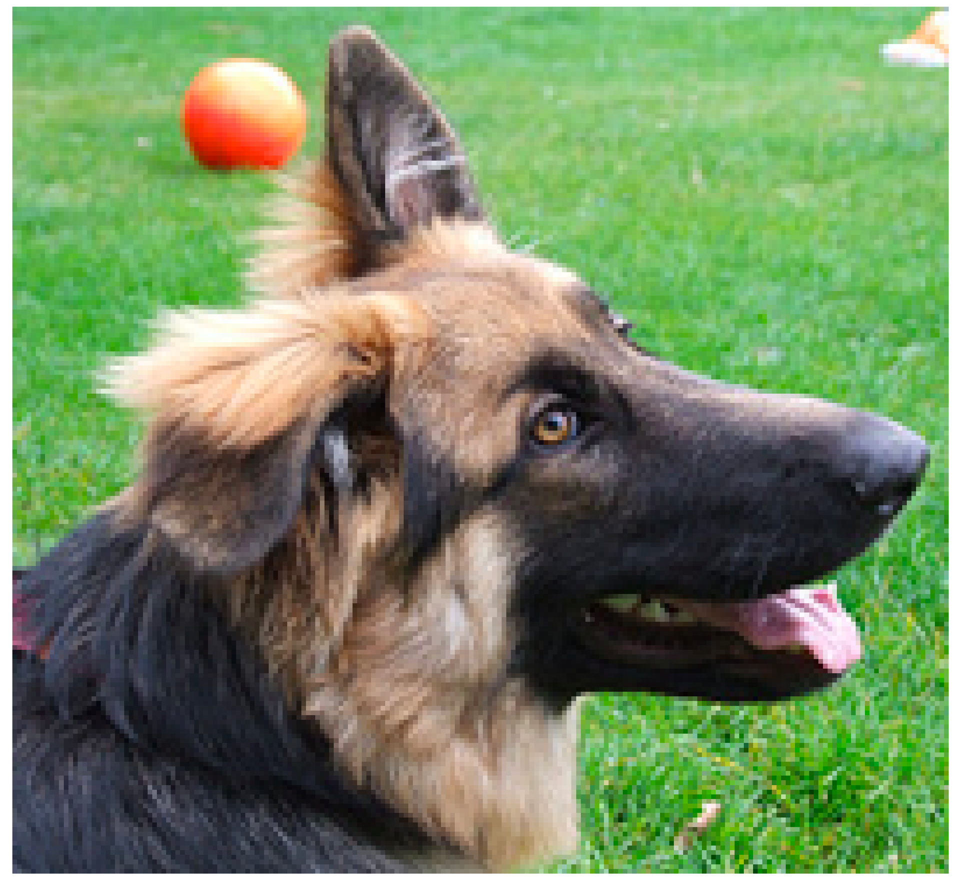


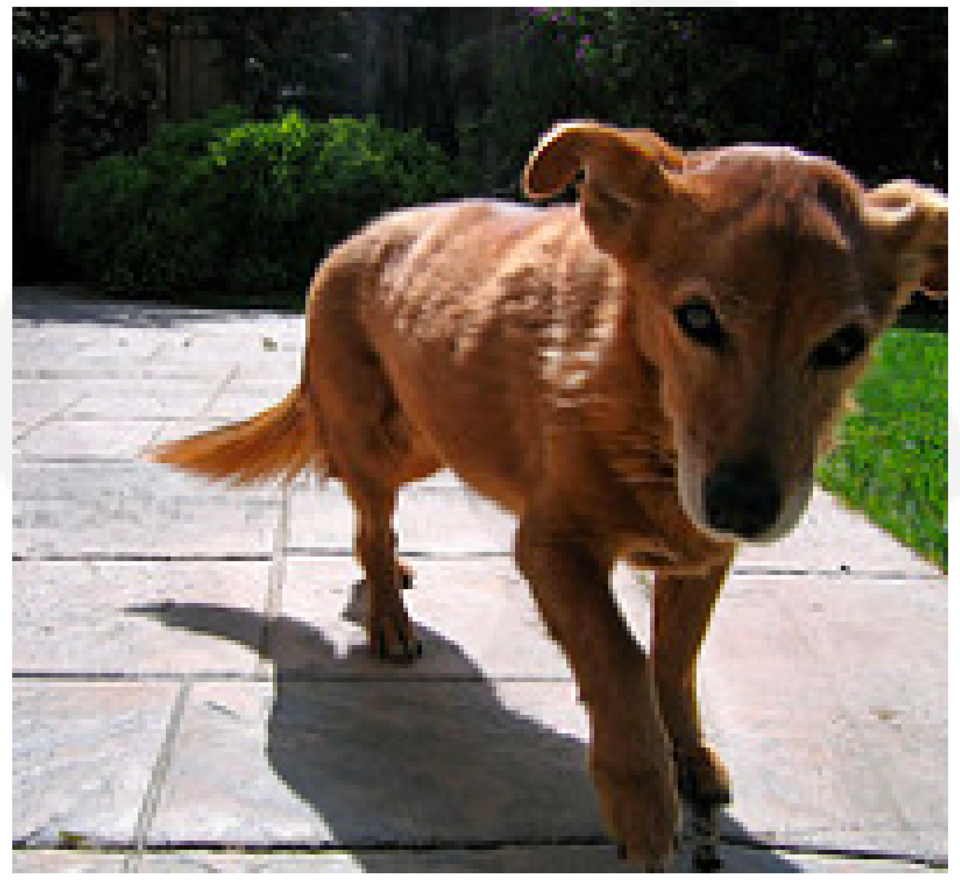
dog

tree

Rethinking Generalization

"Understanding Deep Neural Networks Requires Rethinking Generalization"



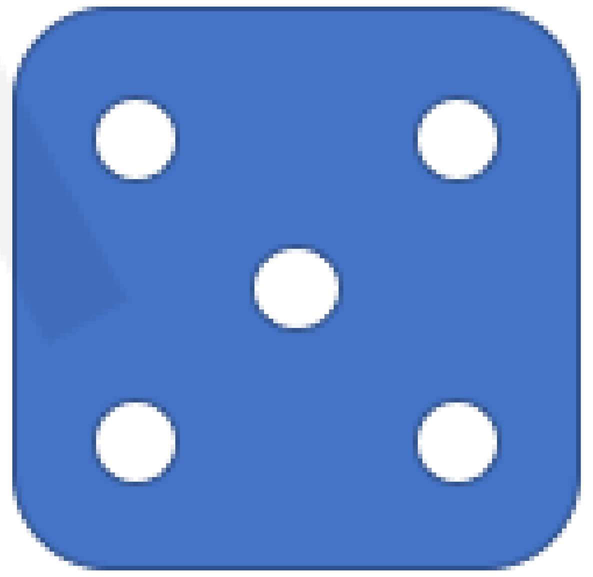


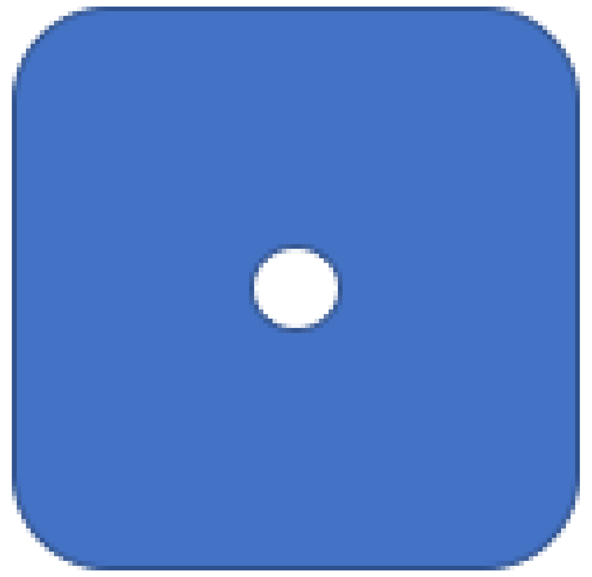
dog

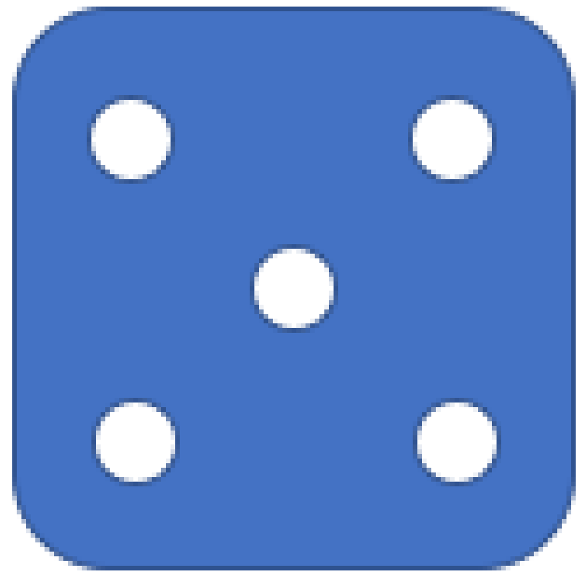
banana

dog

tree

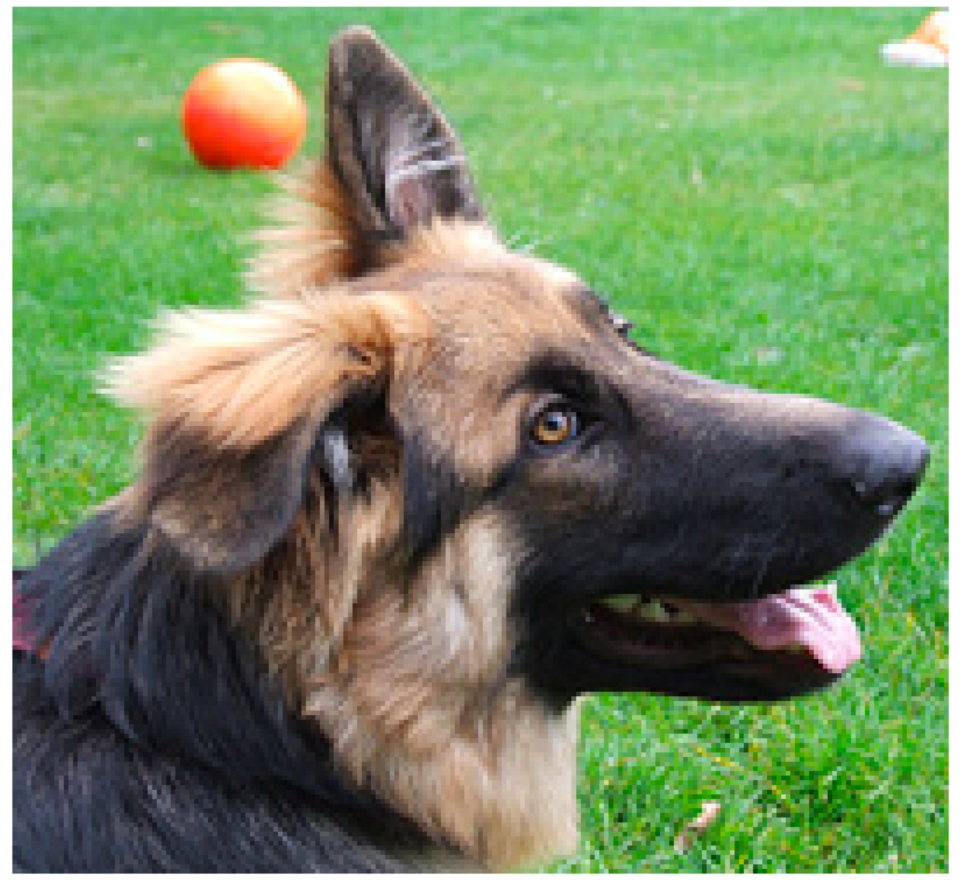






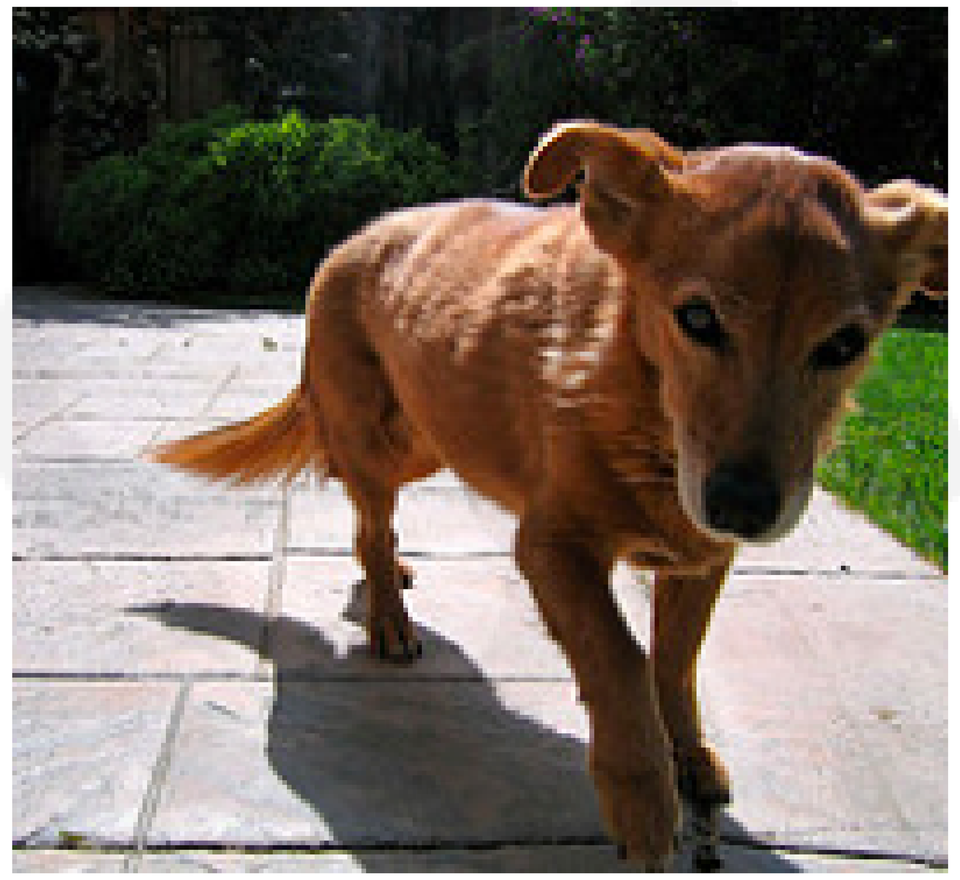
Rethinking Generalization

"Understanding Deep Neural Networks Requires Rethinking Generalization"



dog

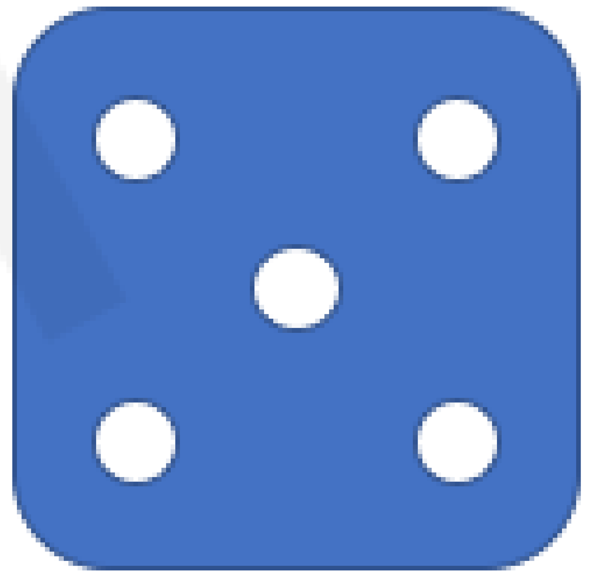
banana



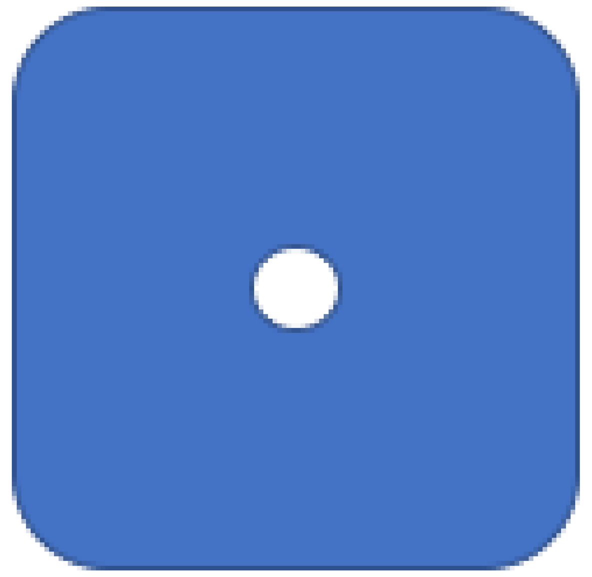
dog

tree

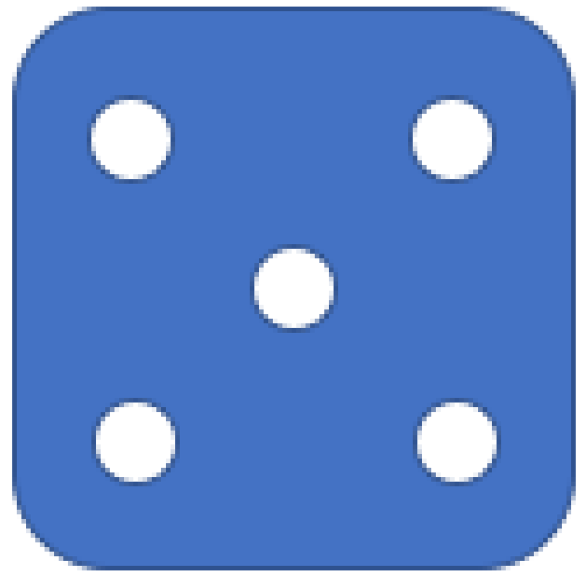
banana



dog



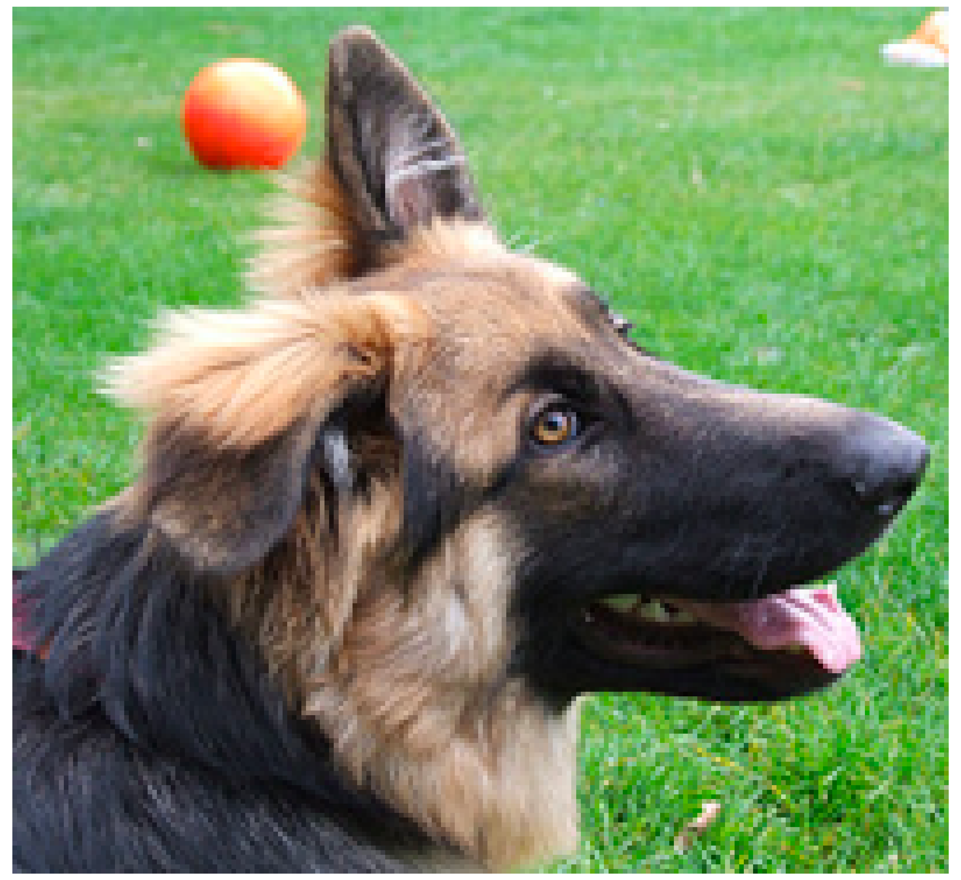
tree



dog

Rethinking Generalization

"Understanding Deep Neural Networks Requires Rethinking Generalization"

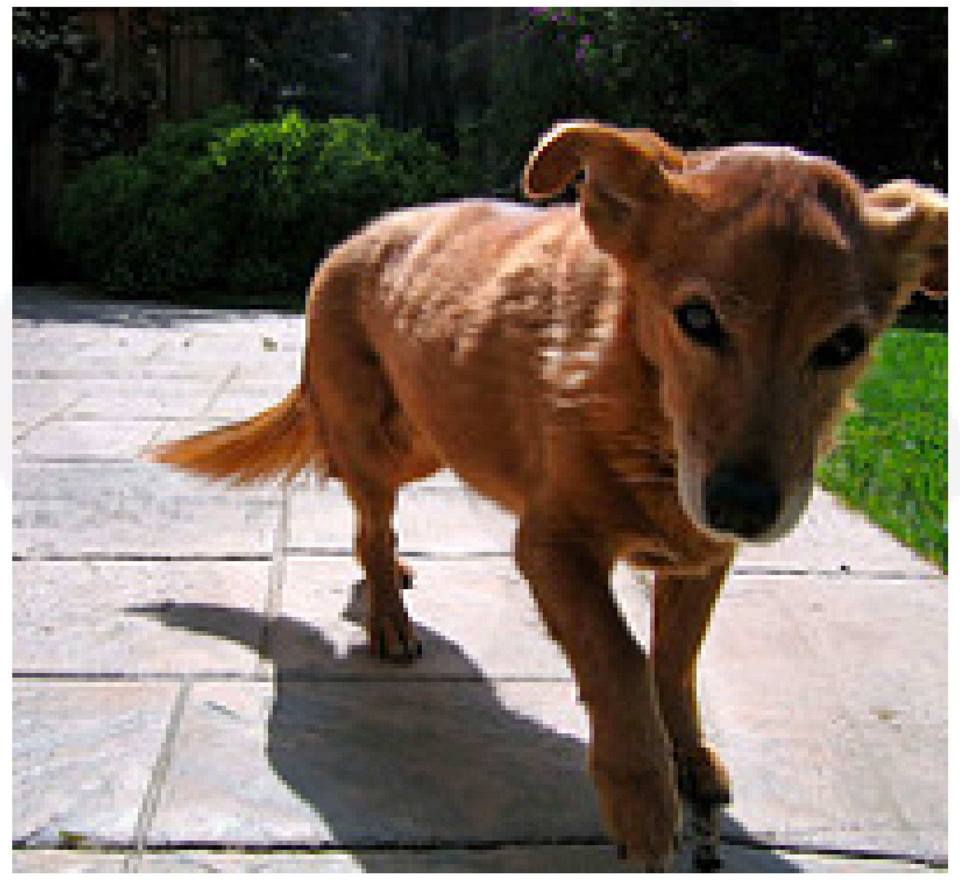


~~dog~~

banana

~~banana~~

dog



~~dog~~

tree

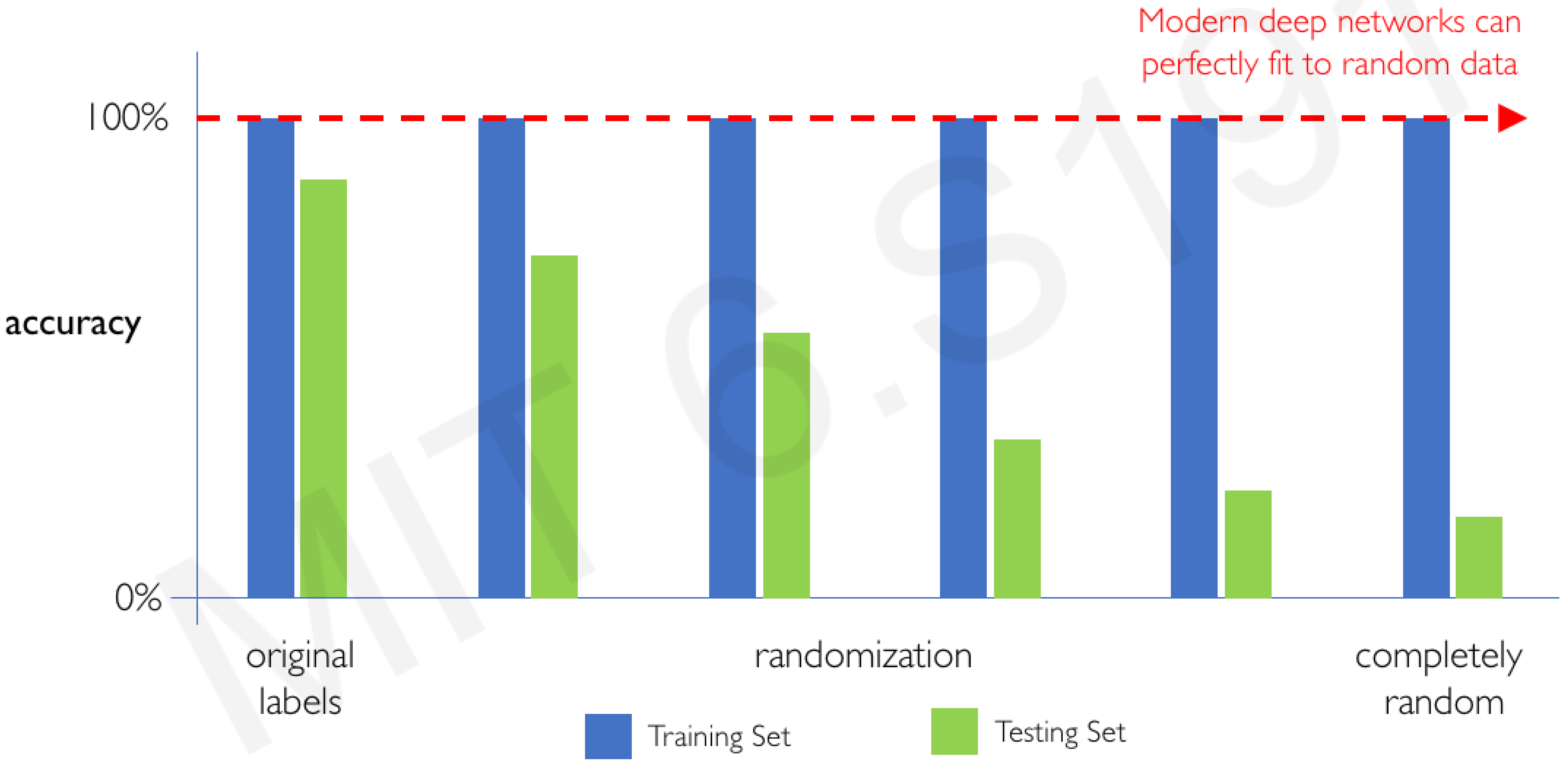
~~tree~~

dog

Capacity of Deep Neural Networks

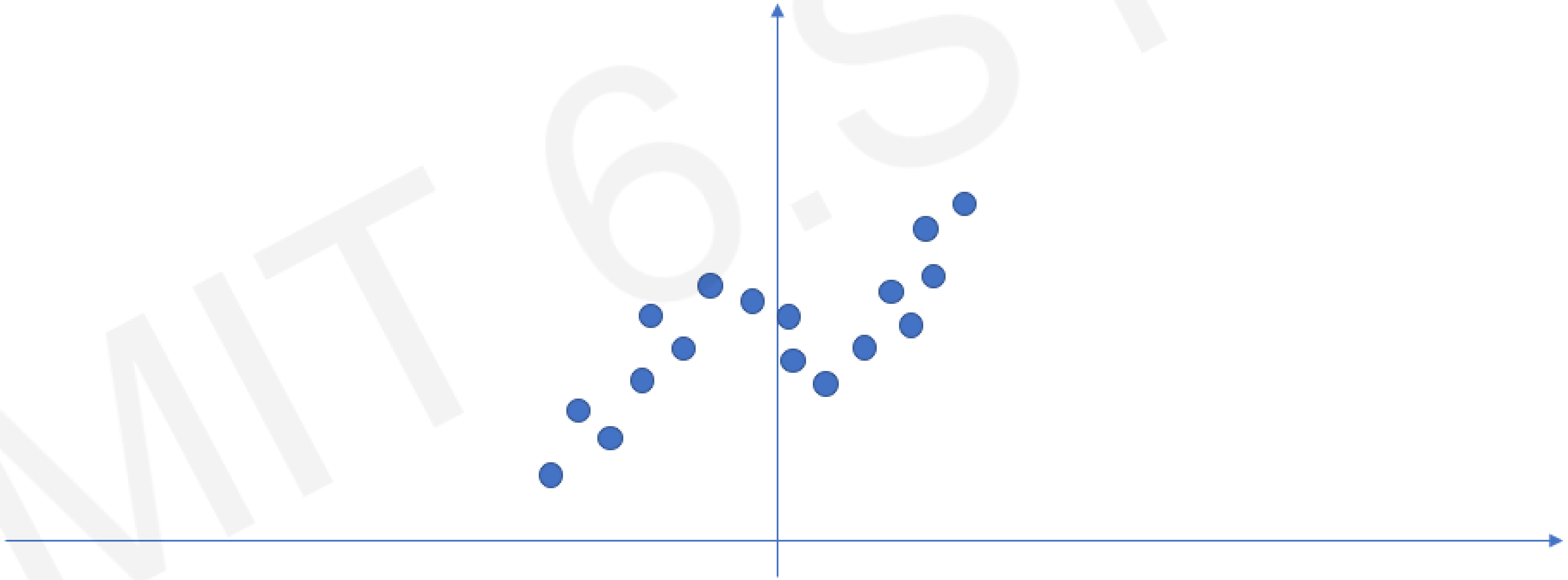
Capacity of Deep Neural Networks

Capacity of Deep Neural Networks



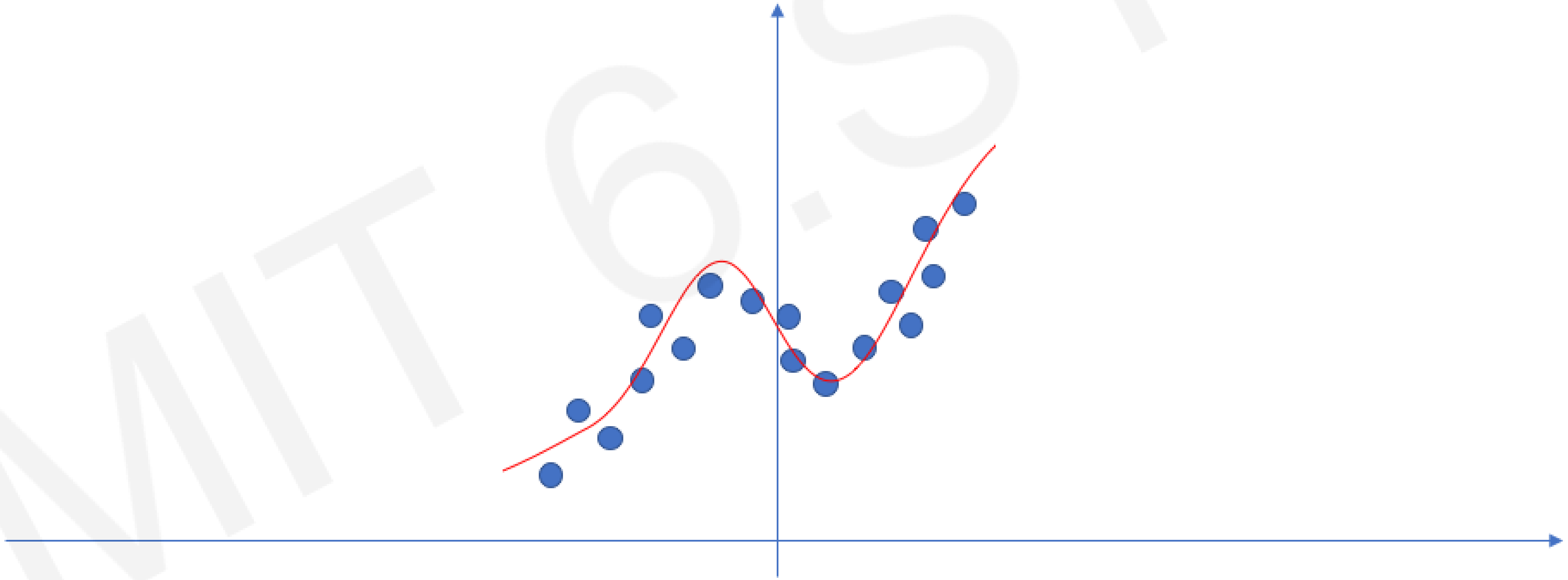
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



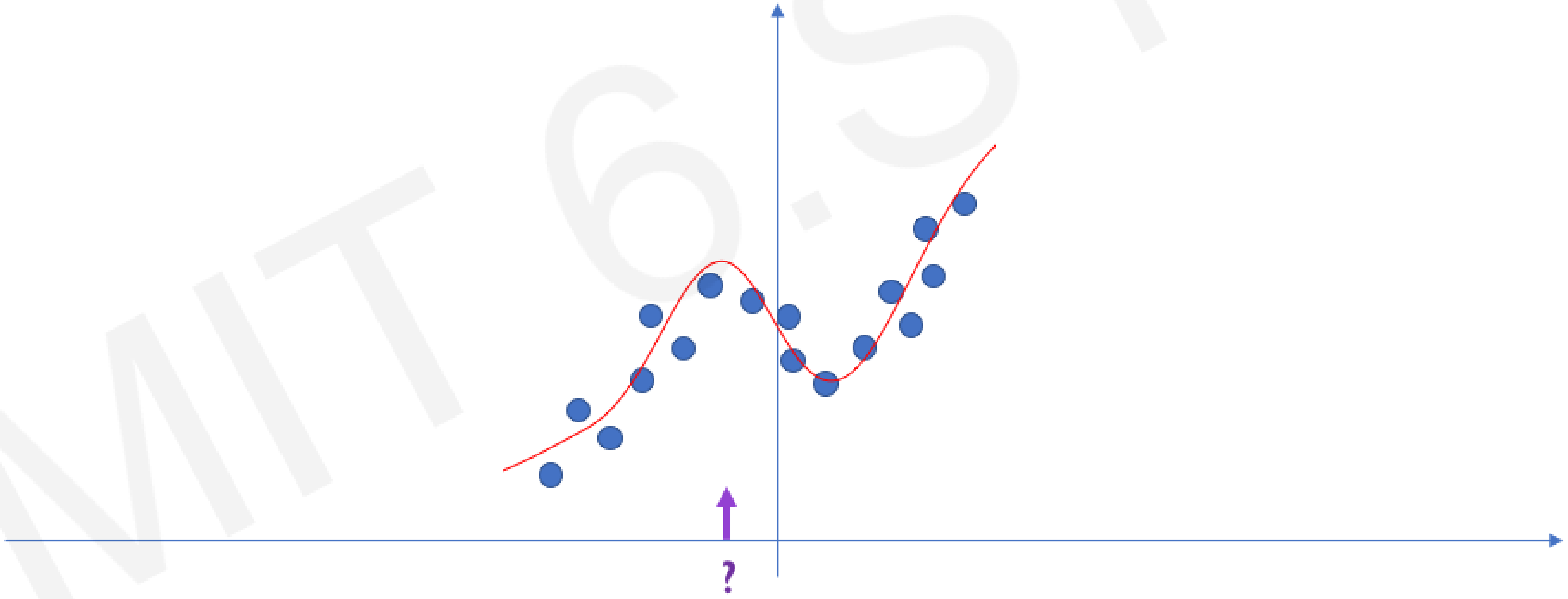
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



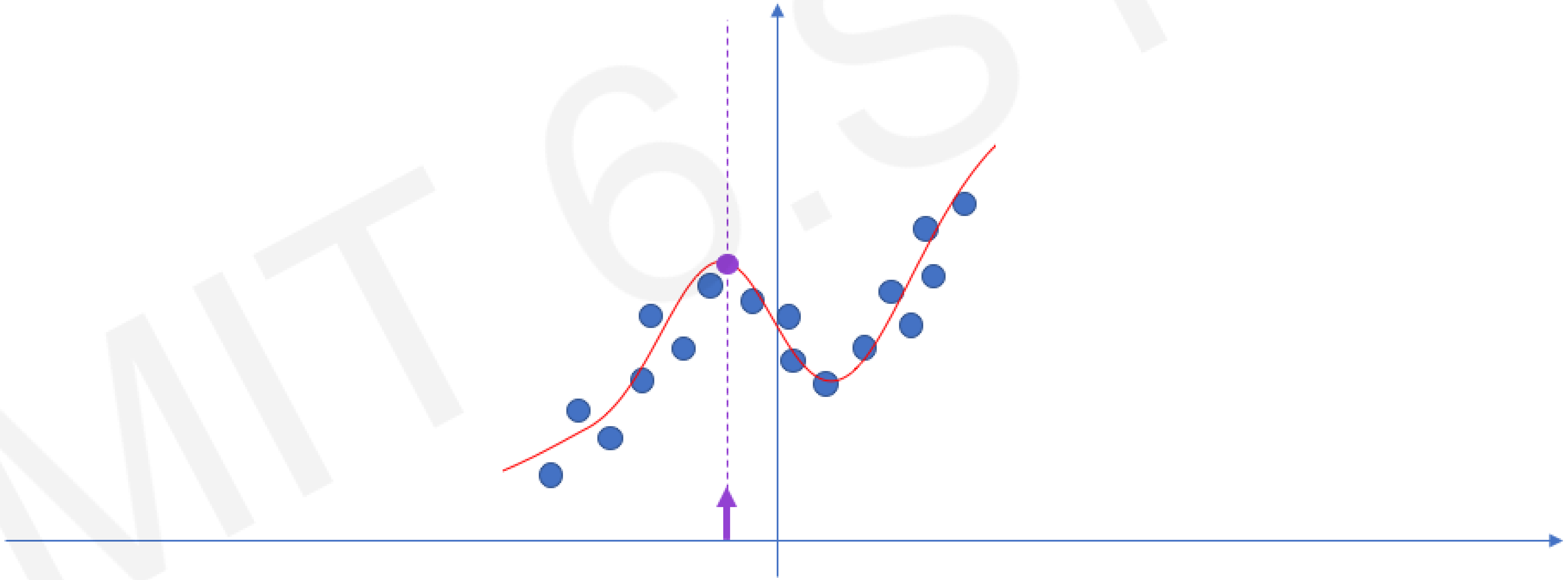
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



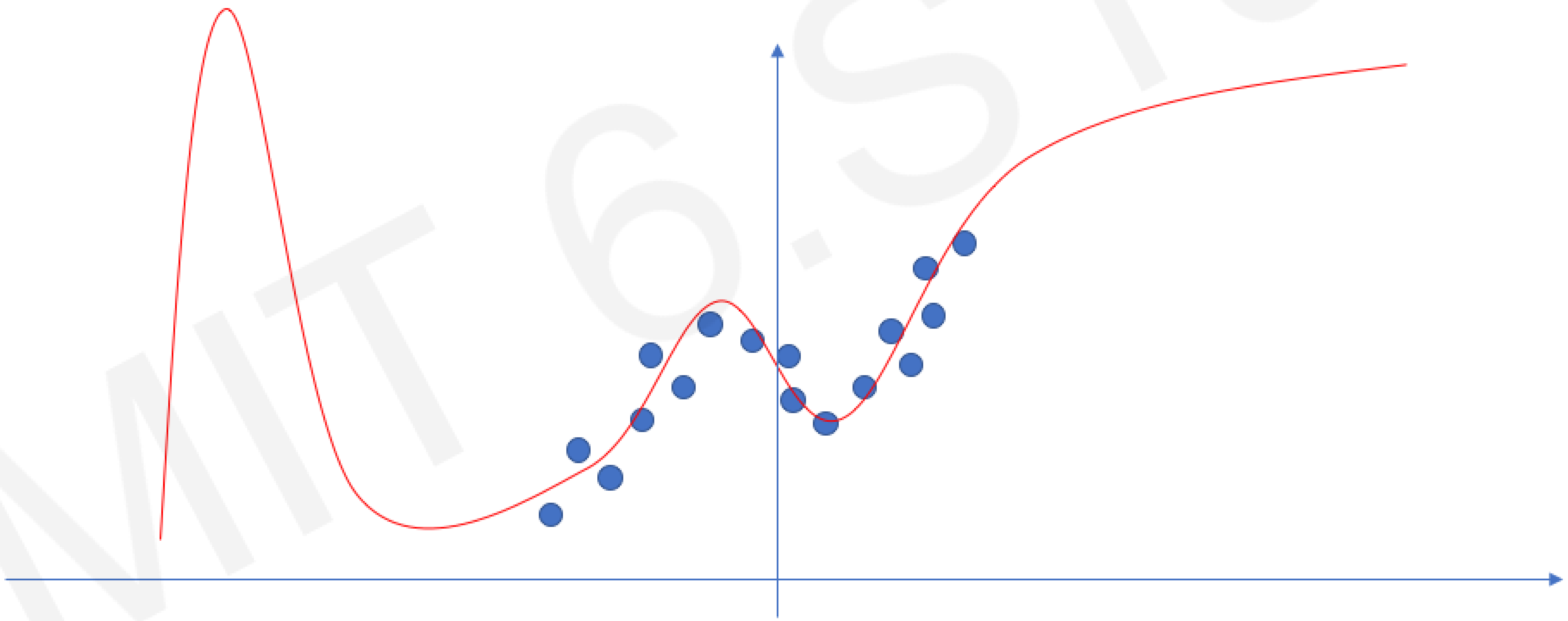
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



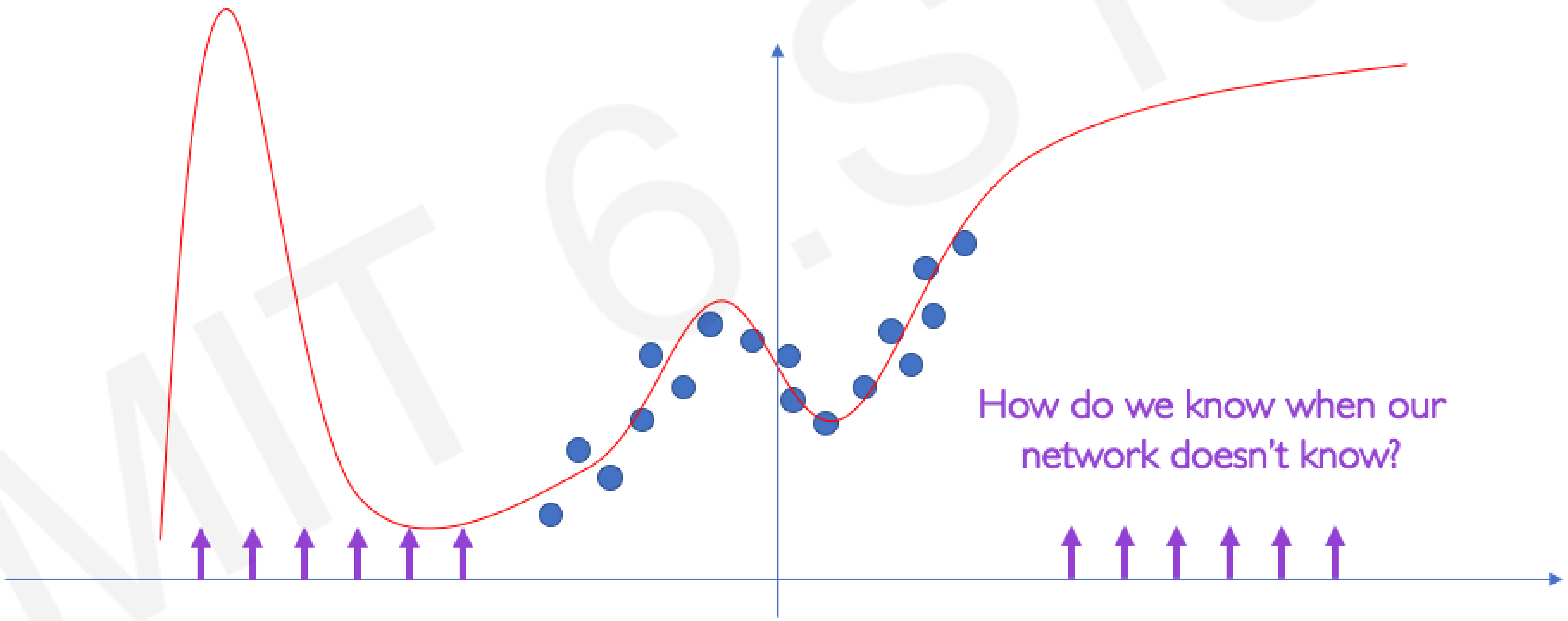
Neural Networks as Function Approximators

Neural networks are **excellent** function approximators



Neural Networks as Function Approximators

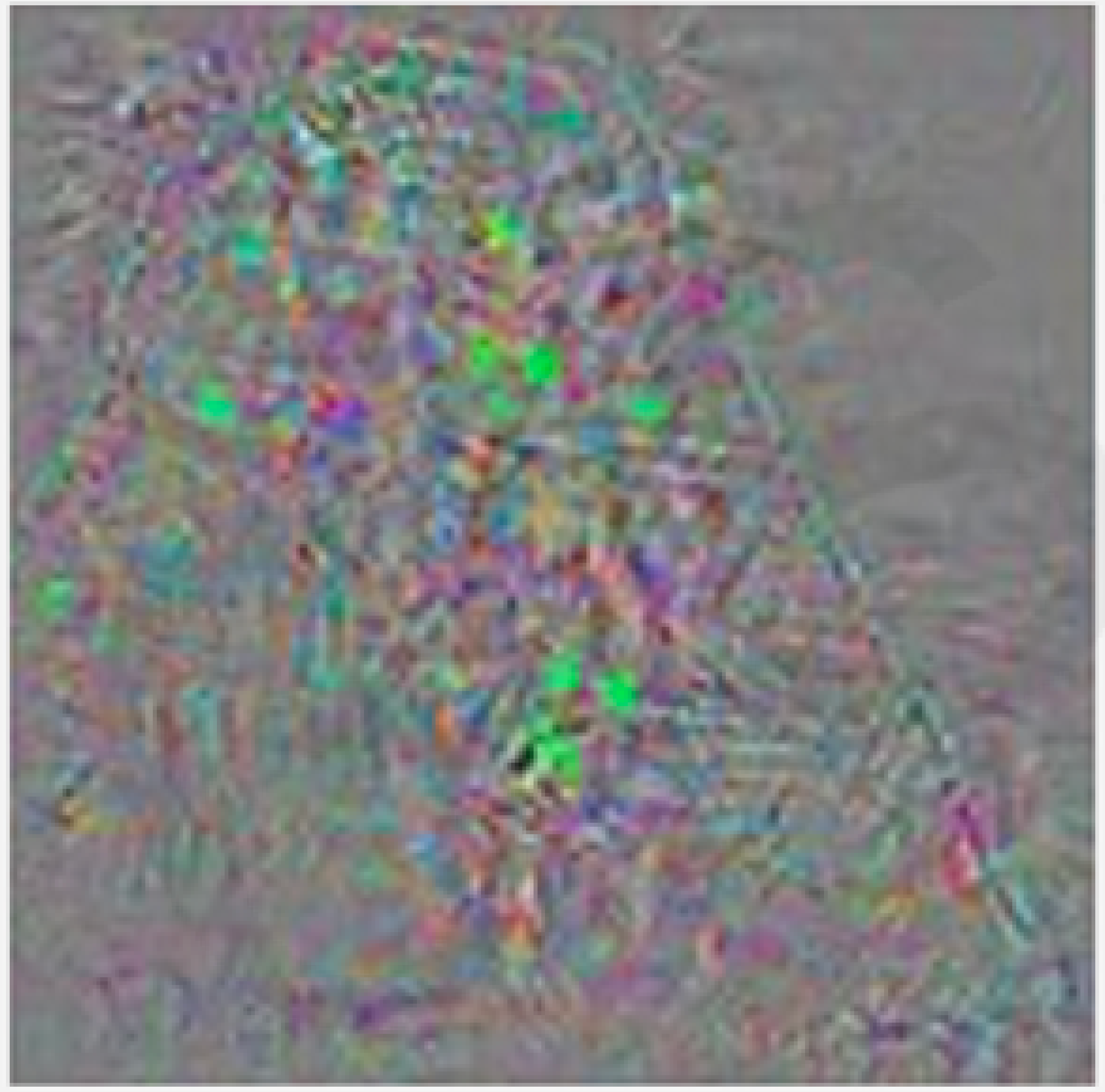
Neural networks are **excellent** function approximators
…when they have training data



Adversarial Attacks on Neural Networks

Original image

Temple (97%)



Perturbations

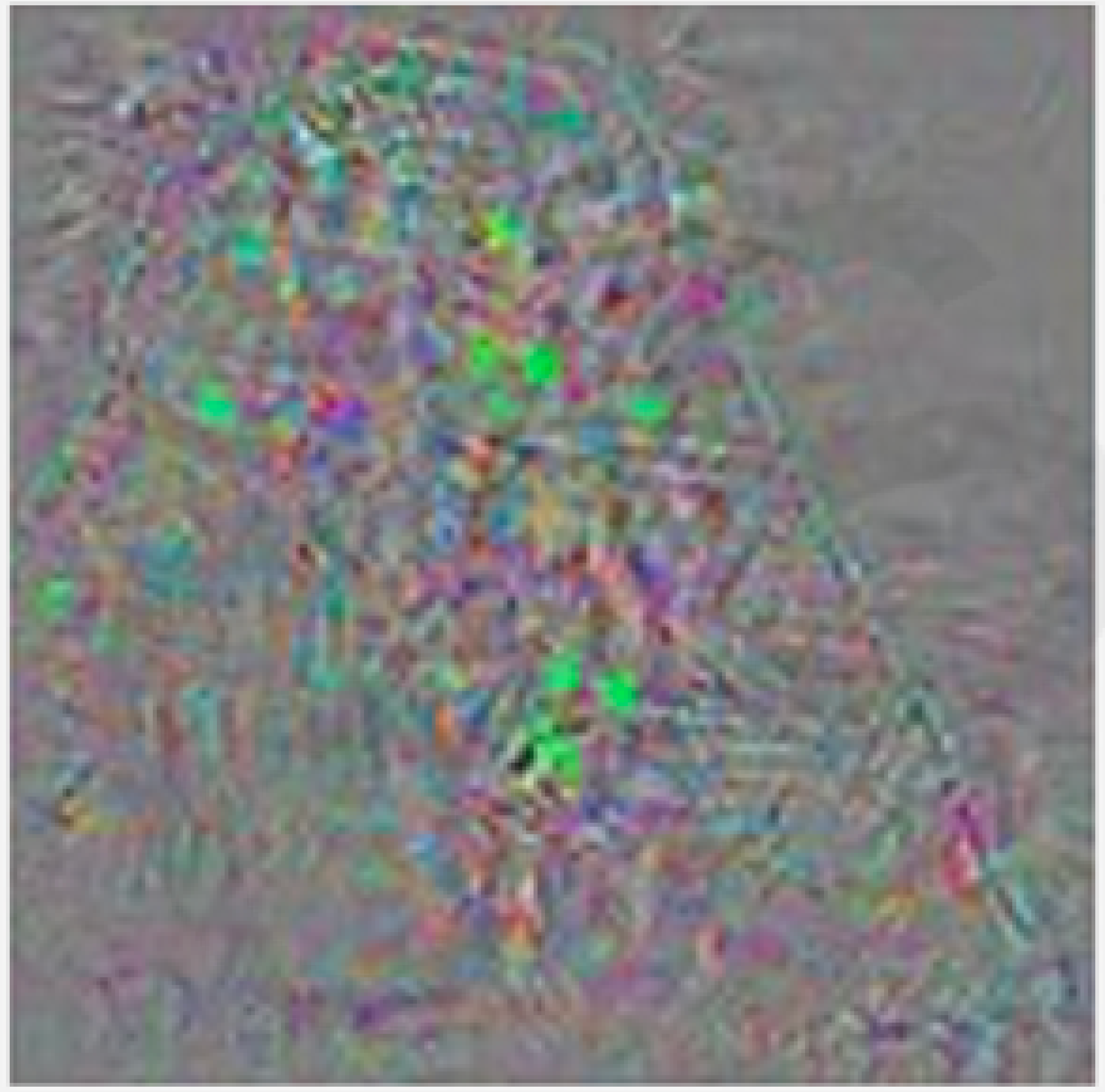
Adversarial example

Ostrich (98%)

Adversarial Attacks on Neural Networks

Original image

Temple (97%)



Perturbations

Adversarial example

Ostrich (98%)

Adversarial Attacks on Neural Networks

Remember:

We train our networks with gradient descent

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

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

Adversarial Attacks on Neural Networks

Remember:

We train our networks with gradient descent

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

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

Adversarial Attacks on Neural Networks

Remember:

We train our networks with gradient descent

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

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

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

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

Adversarial Attacks on Neural Networks

Adversarial Image:

Modify image to increase error

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

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

Adversarial Attacks on Neural Networks

Adversarial Image:

Modify image to increase error

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

Fix your weights θ, and true label y

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

Synthesizing Robust Adversarial Examples

Neural Network Limitations…

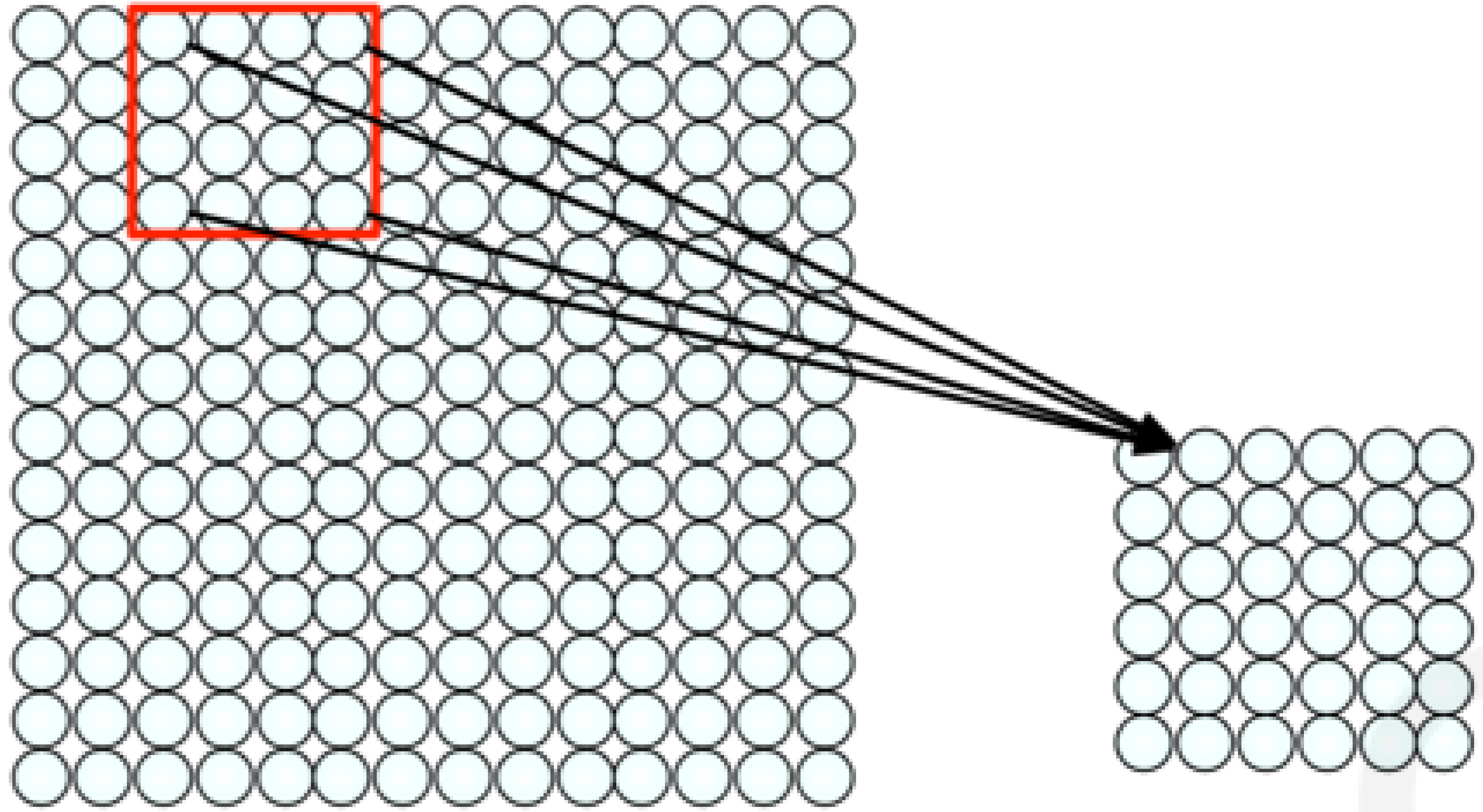
- Very **data hungry** (eg. often millions of examples)
- **Computationally intensive** to train and deploy (tractably requires GPUs)
- Easily fooled by **adversarial examples**
- Can be subject to **algorithmic bias**
- Difficult to **encode structure** and prior knowledge during learning
- Poor at **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

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**
- Difficult to **encode structure** and prior knowledge during learning
- 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

New Frontiers I:
Encoding Structure into Deep Learning

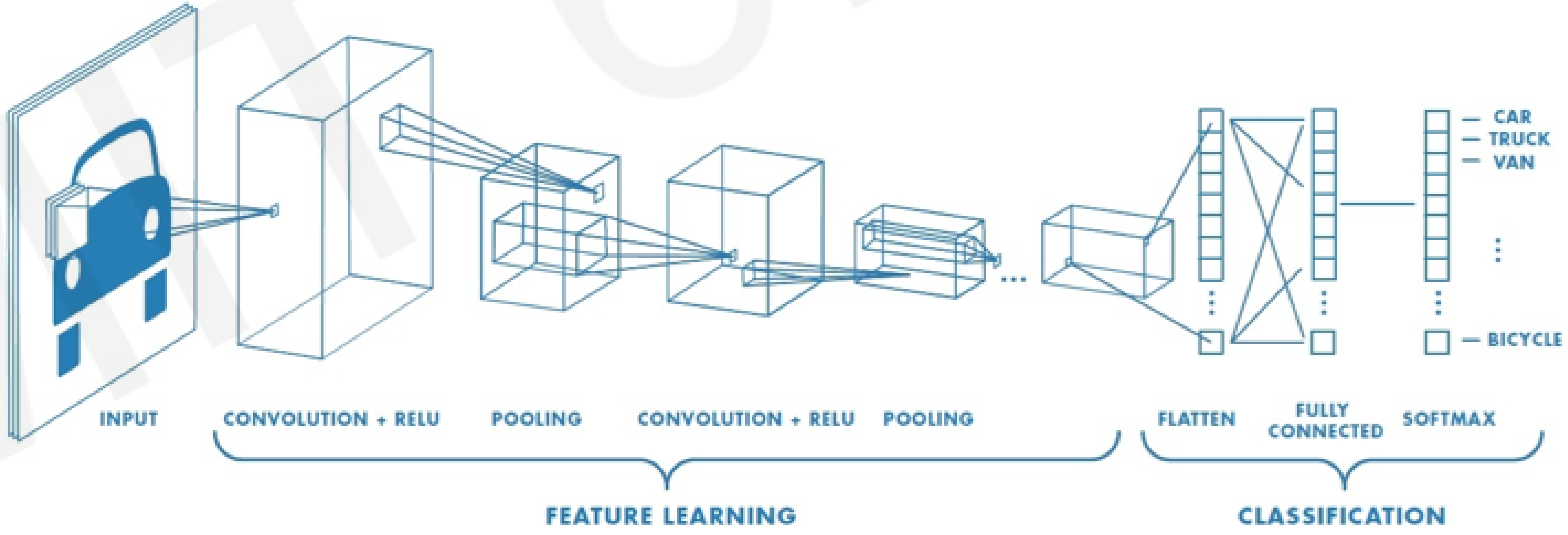
CNNs: Using Spatial Structure



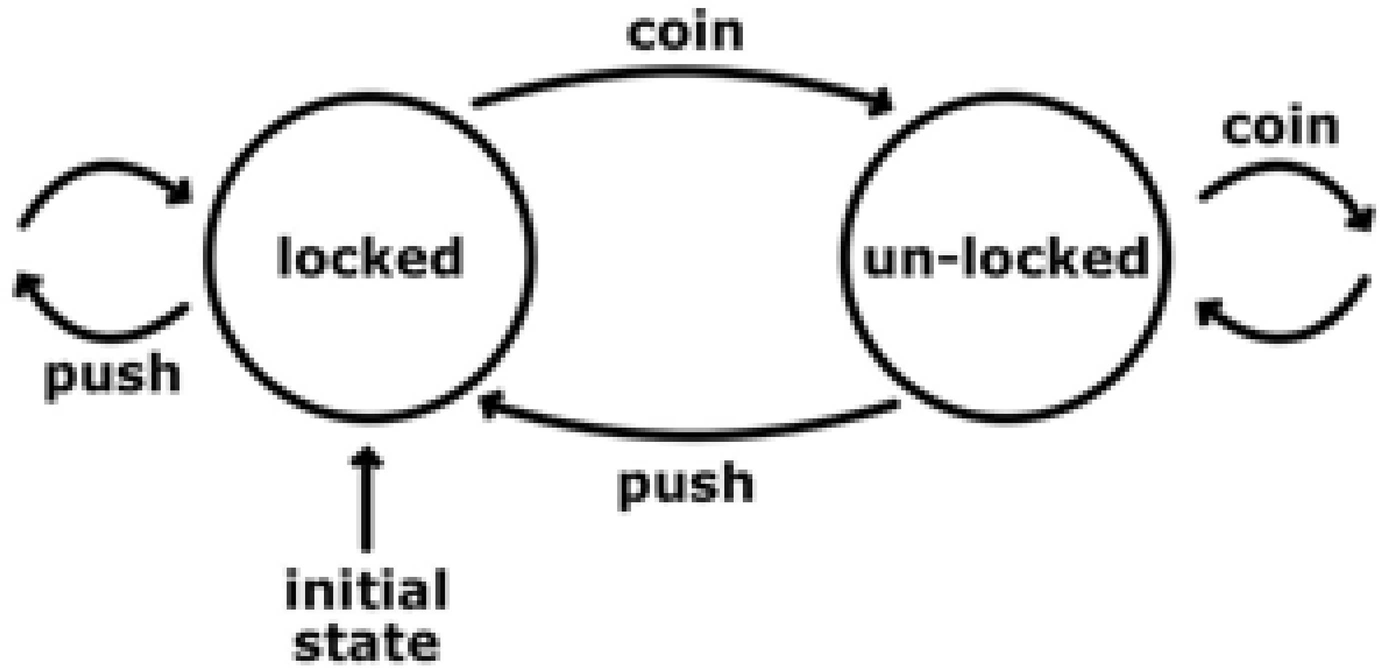
1) Apply a set of weights to extract **local features**

2) Use **multiple filters** to extract different features

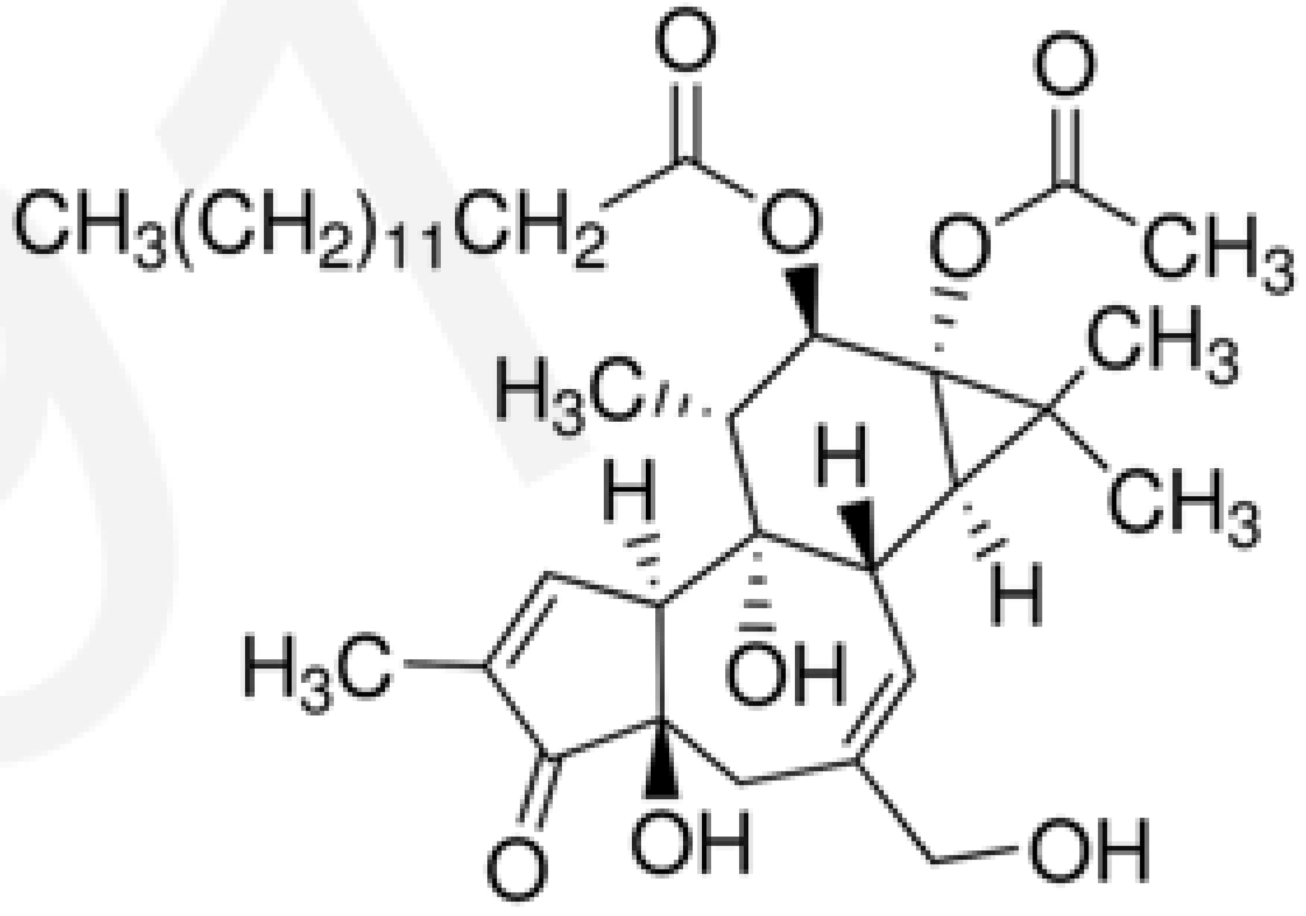
3) **Spatially share** parameters of each filter



Graphs as a Structure for Representing Data

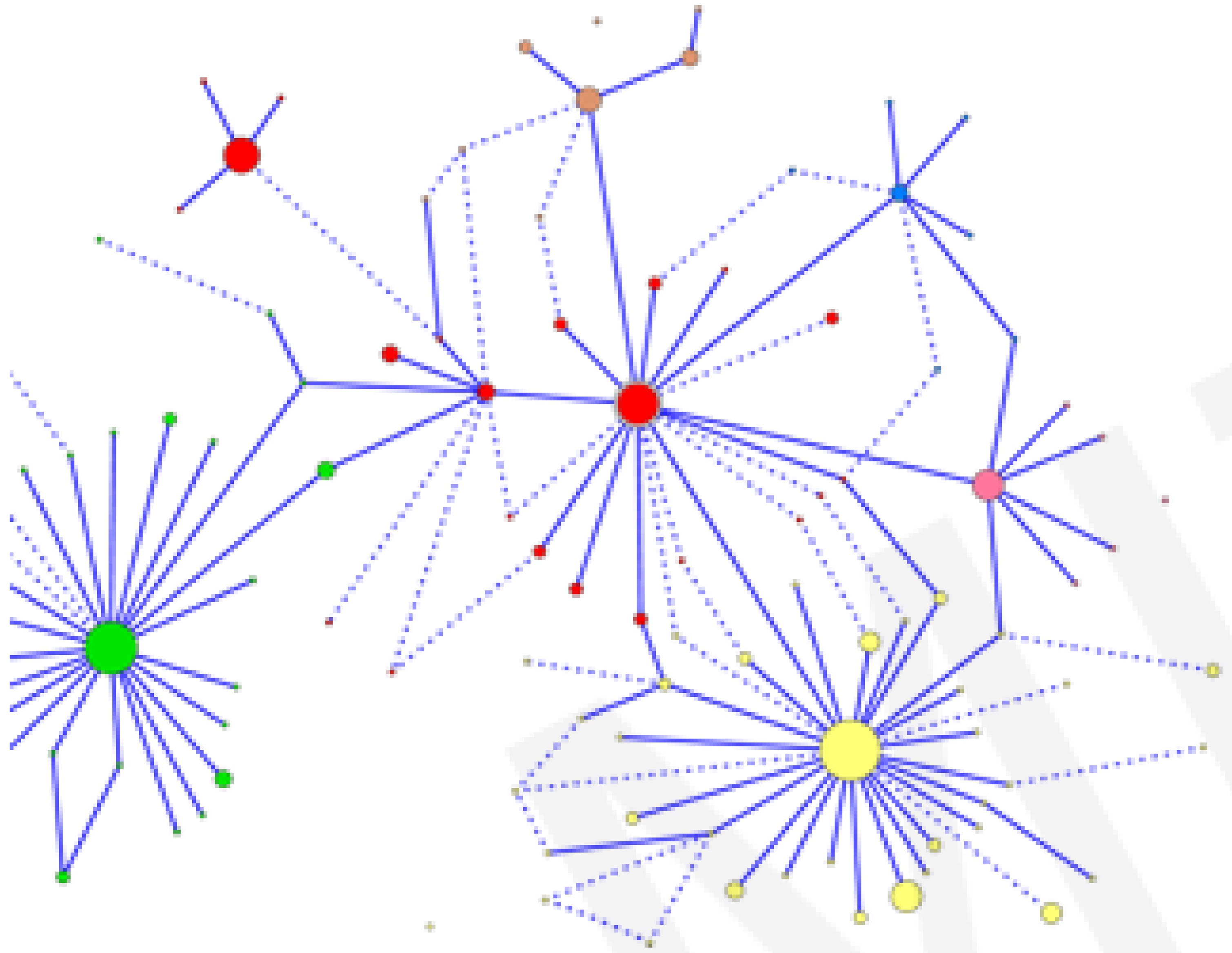


State Machines



Molecules

Social Networks



Biological Networks

Mobility & Transport

Graph Convolutional Networks

Convolutional Networks

Graph Convolutional Networks (GCNs)

Graph Convolutional Networks

Convolutional Networks

Graph Convolutional Networks (GCNs)

Graph Convolutional Networks

Convolutional Networks

Graph Convolutional Networks (GCNs)

Graph Convolutional Networks

Convolutional Networks

Graph Convolutional Networks (GCNs)

Graph Convolutional Networks

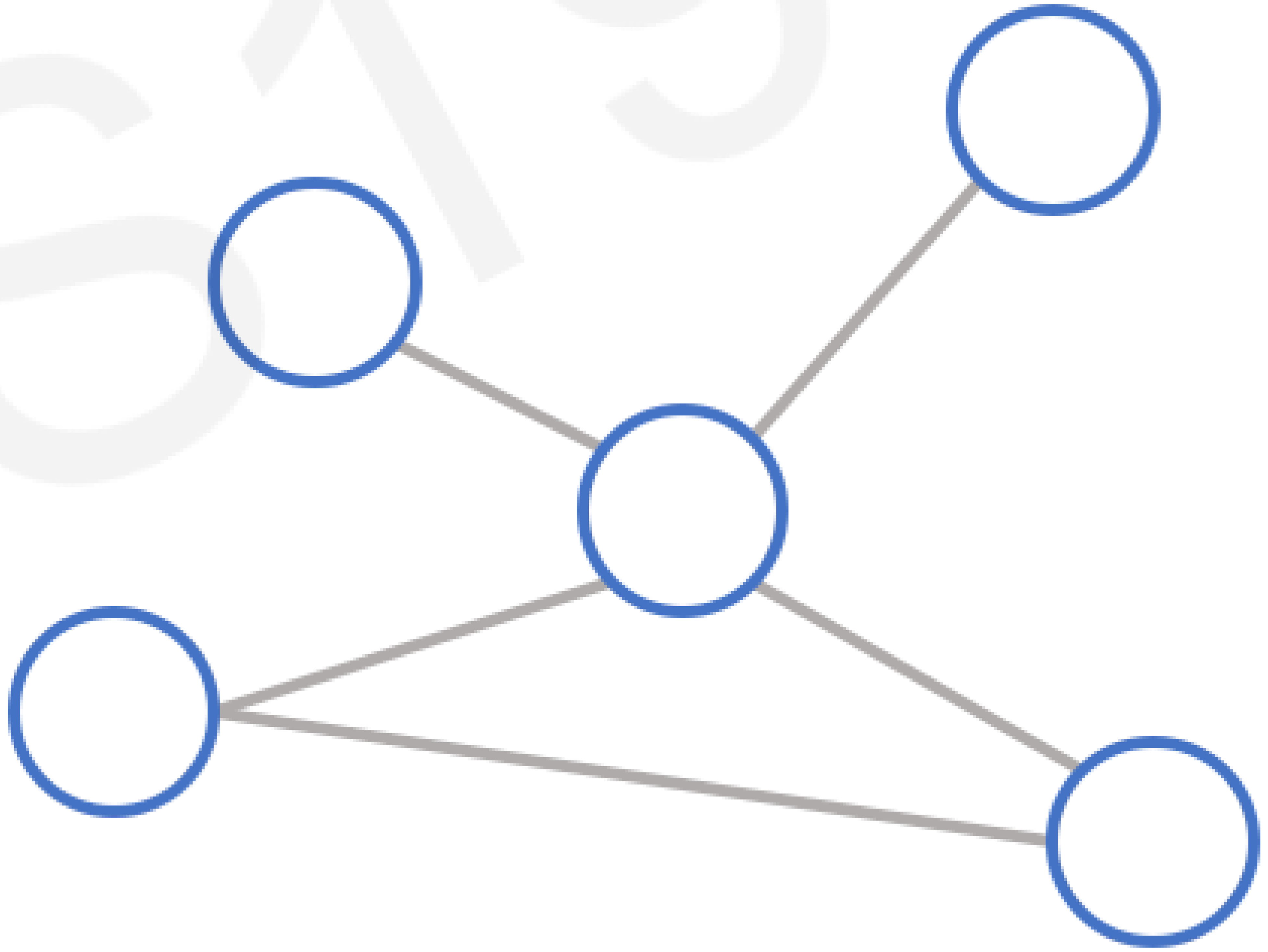
Convolutional Networks

Graph Convolutional Networks (GCNs)

Graph Convolutional Networks

Convolutional Networks

Graph Convolutional Networks (GCNs)

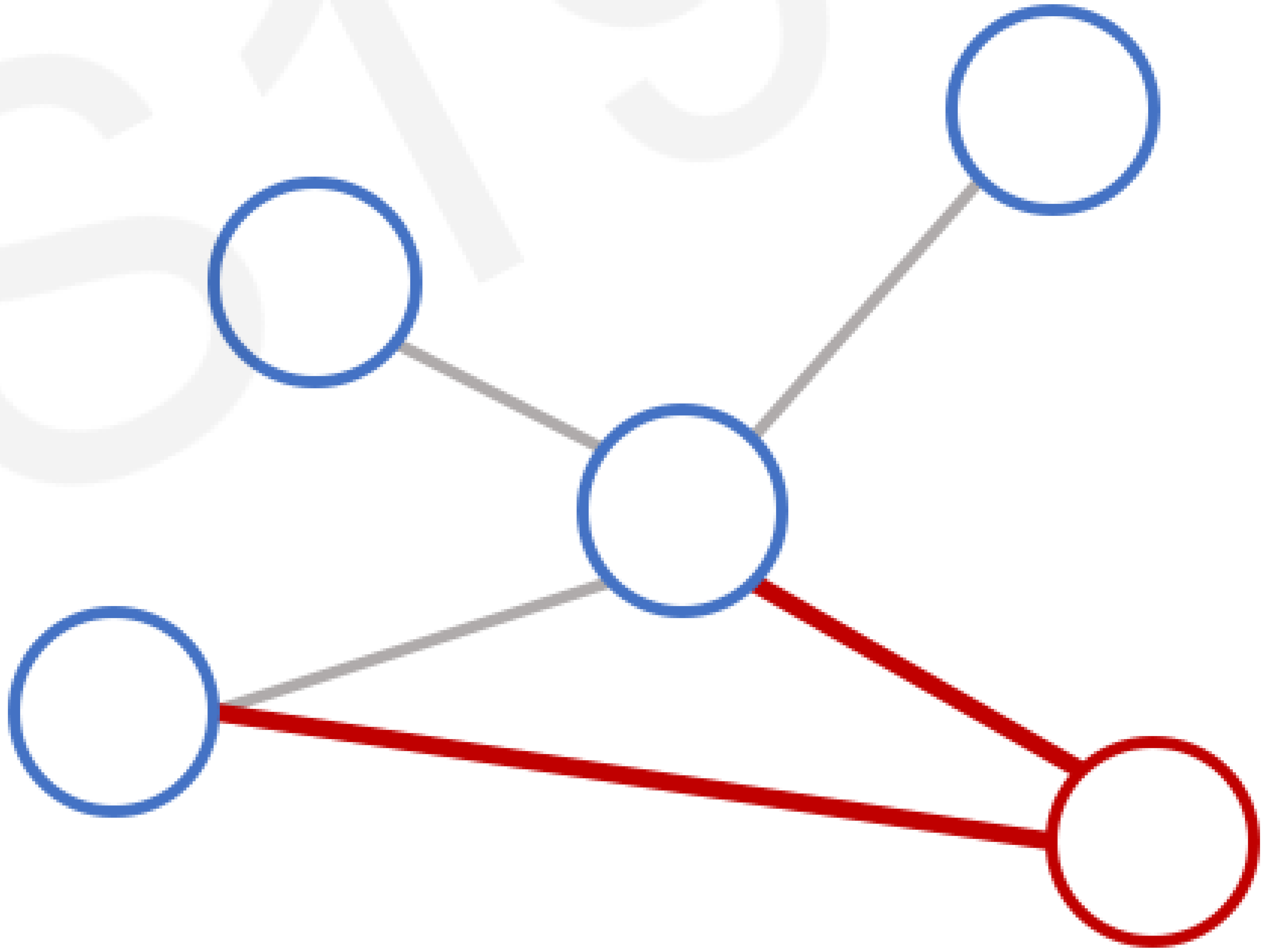


Duvenaud+ *NeurIPS* 2015; Kipf & Welling *ICLR* 2017. 1/29/20

Graph Convolutional Networks

Convolutional Networks

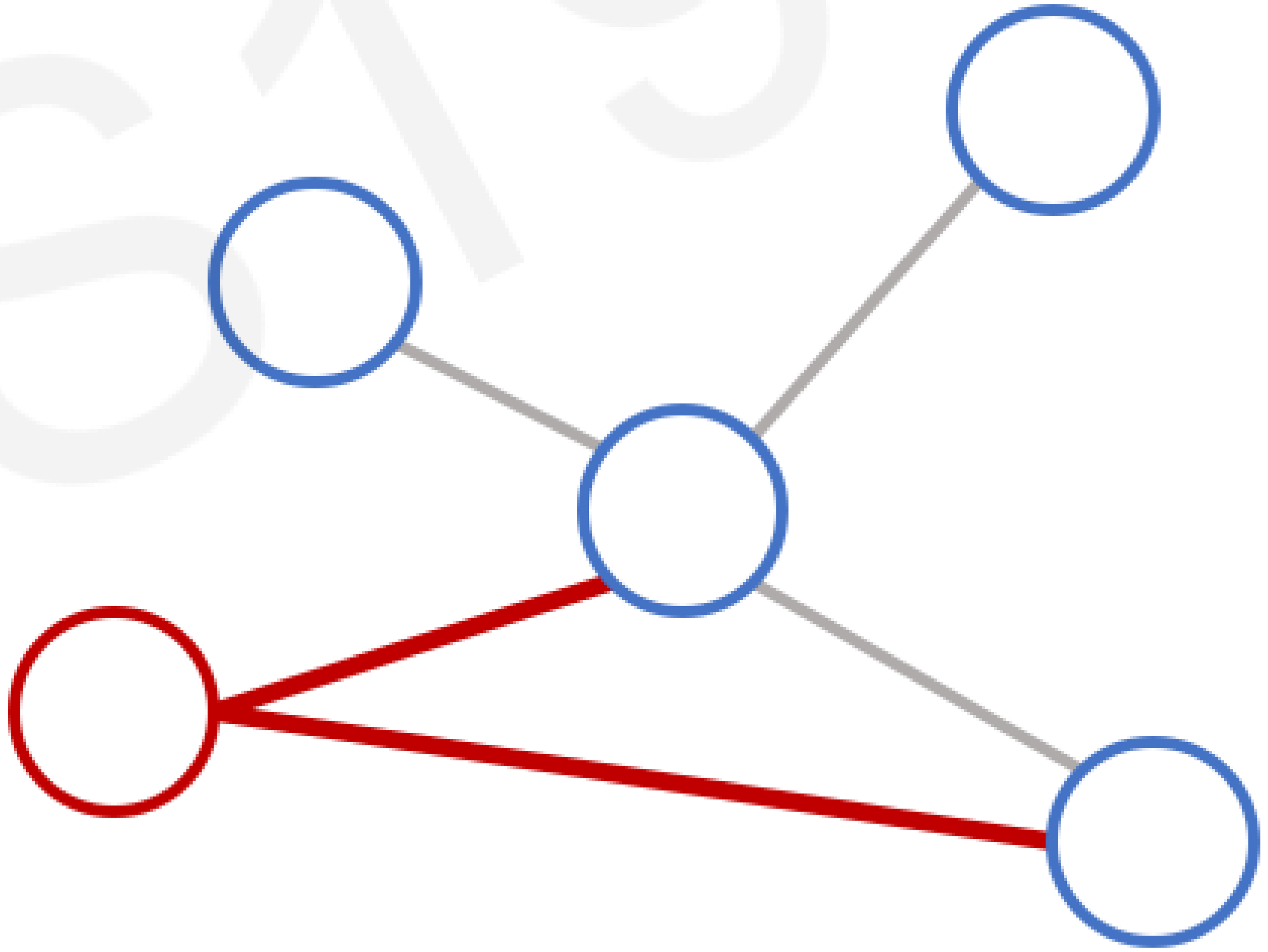
Graph Convolutional Networks (GCNs)



Graph Convolutional Networks

Convolutional Networks

Graph Convolutional Networks (GCNs)

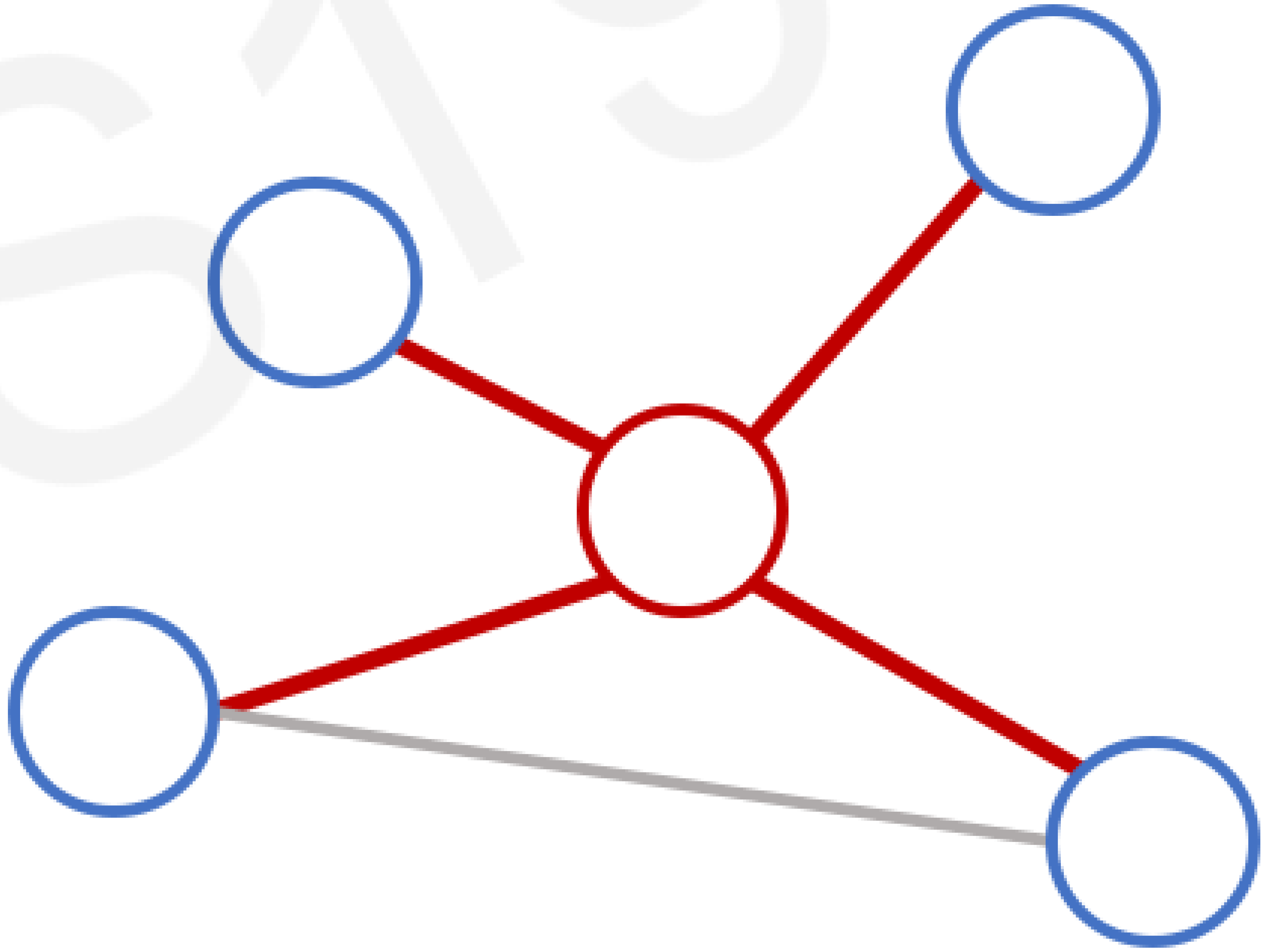


Duvenaud+ *NeurIPS* 2015; Kipf & Welling *ICLR* 2017. 1/29/20

Graph Convolutional Networks

Convolutional Networks

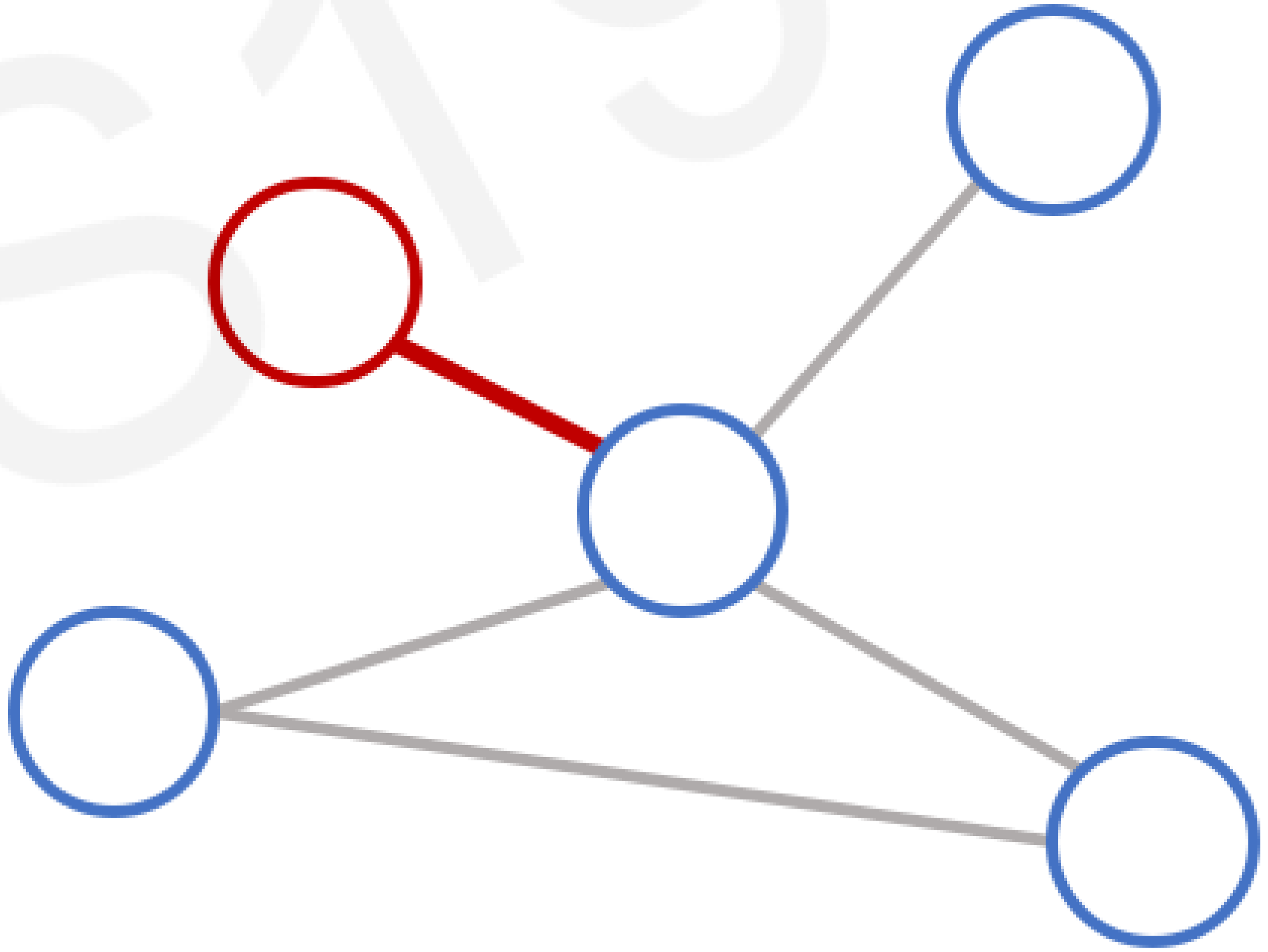
Graph Convolutional Networks (GCNs)



Graph Convolutional Networks

Convolutional Networks

Graph Convolutional Networks (GCNs)

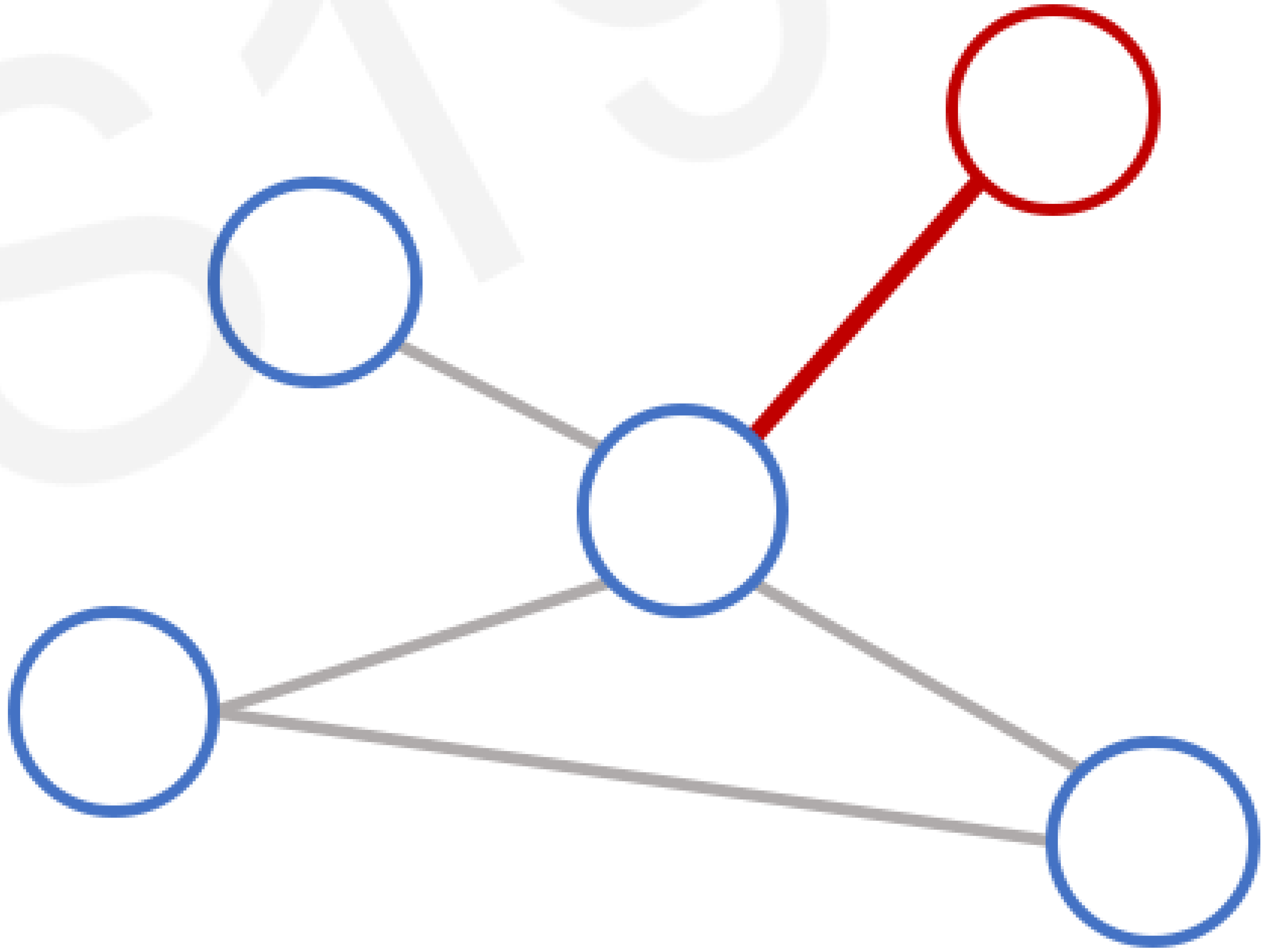


Duvenaud+ *NeurIPS* 2015; Kipf & Welling *ICLR* 2017. 1/29/20

Graph Convolutional Networks

Convolutional Networks

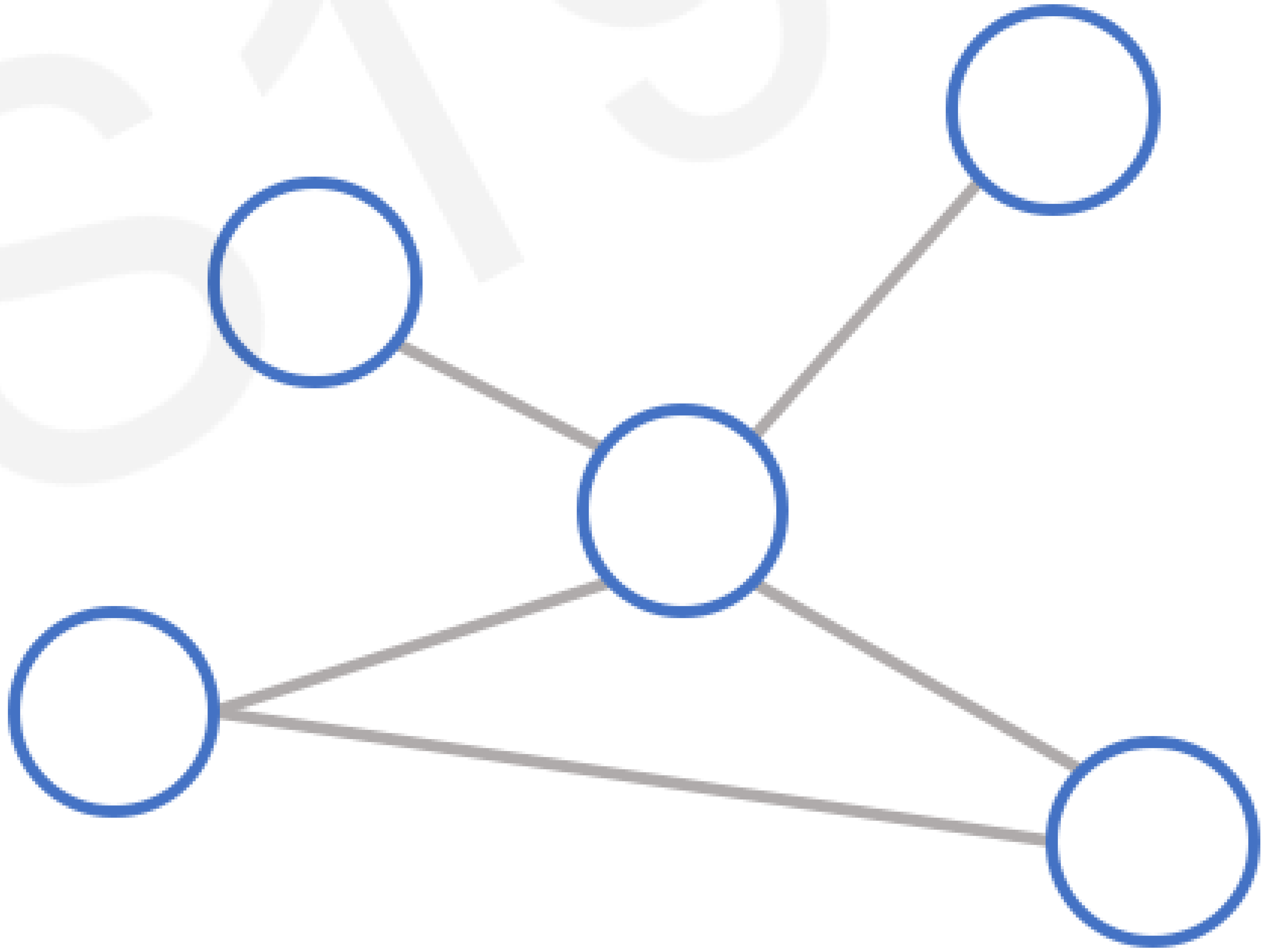
Graph Convolutional Networks (GCNs)



Graph Convolutional Networks

Convolutional Networks

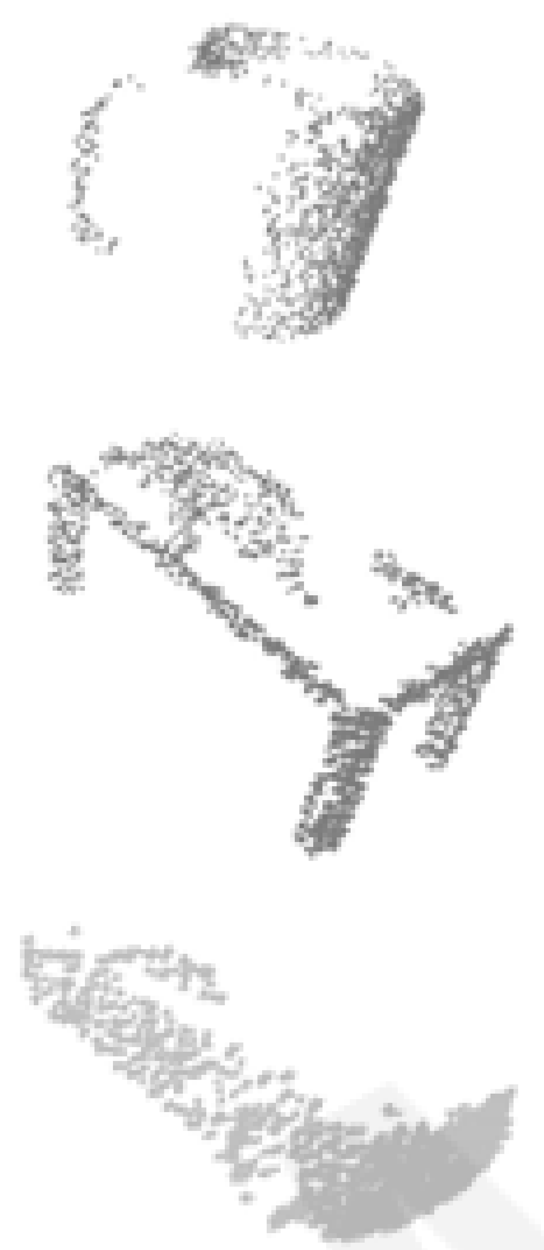
Graph Convolutional Networks (GCNs)



Friday: Graph neural networks for odor prediction
Alex Wiltschko, Google Brain

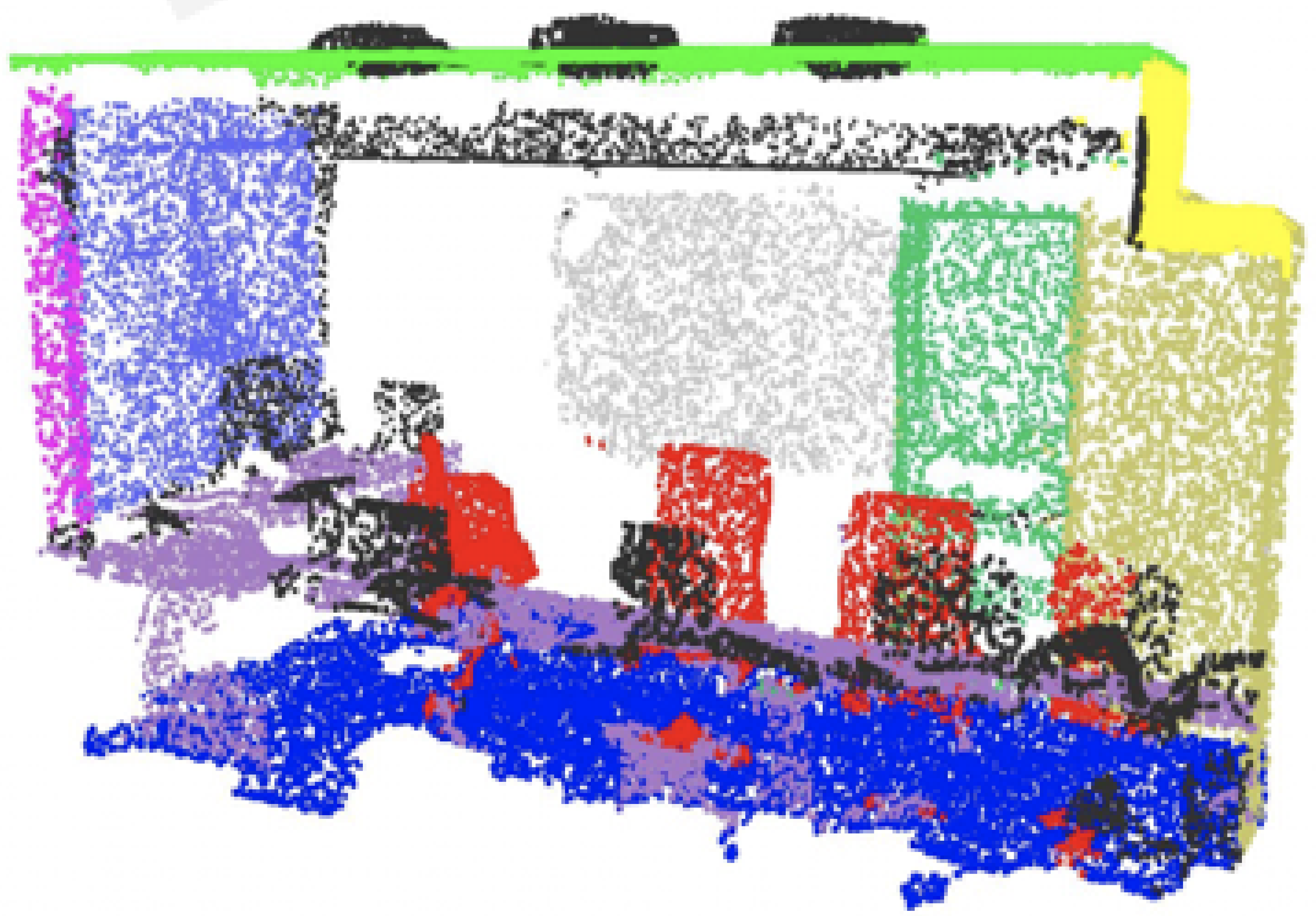
Learning From 3D Data

Point clouds are **unordered sets** with **spatial dependence** between points



Classification

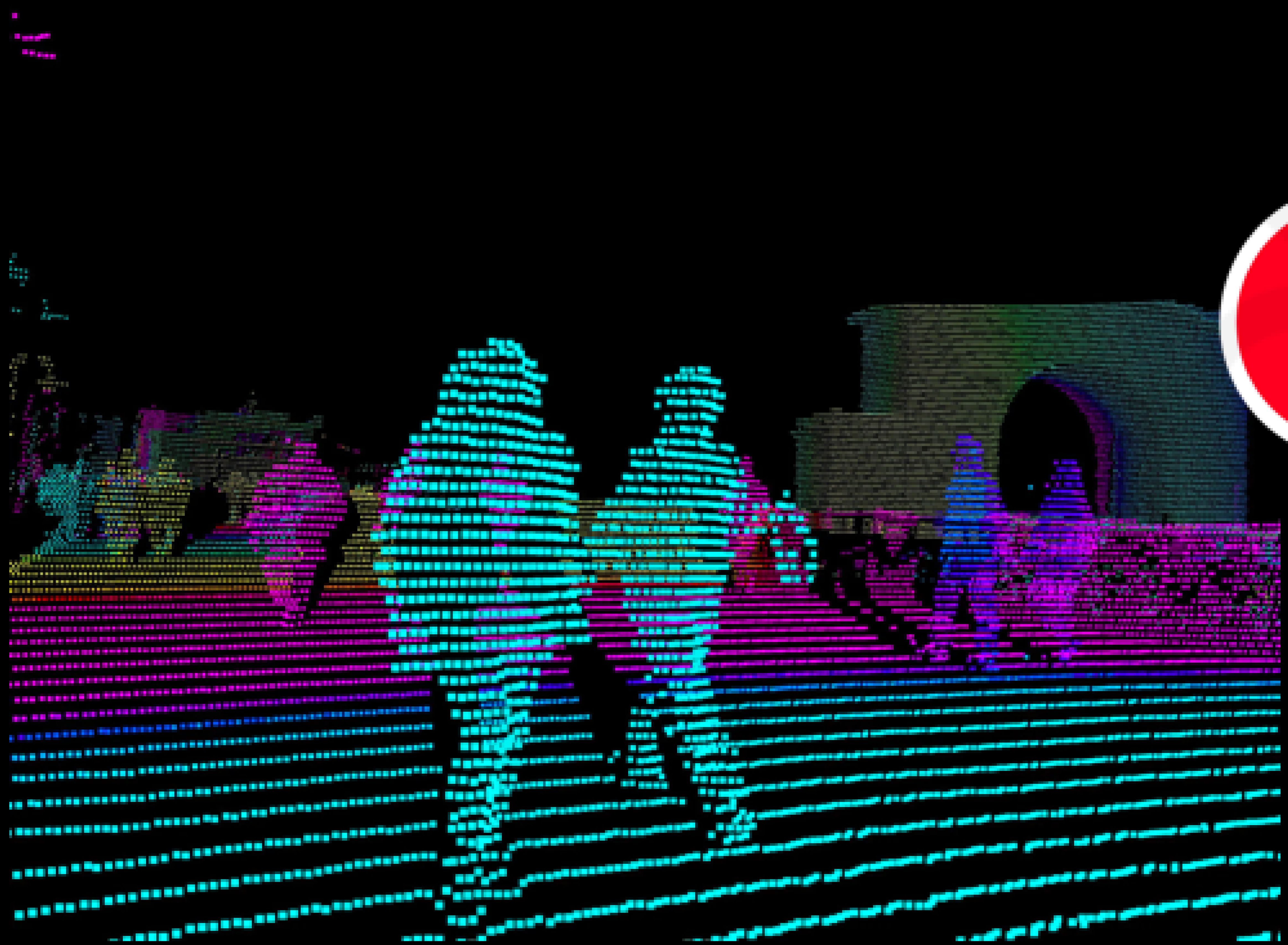
Part Segmentation

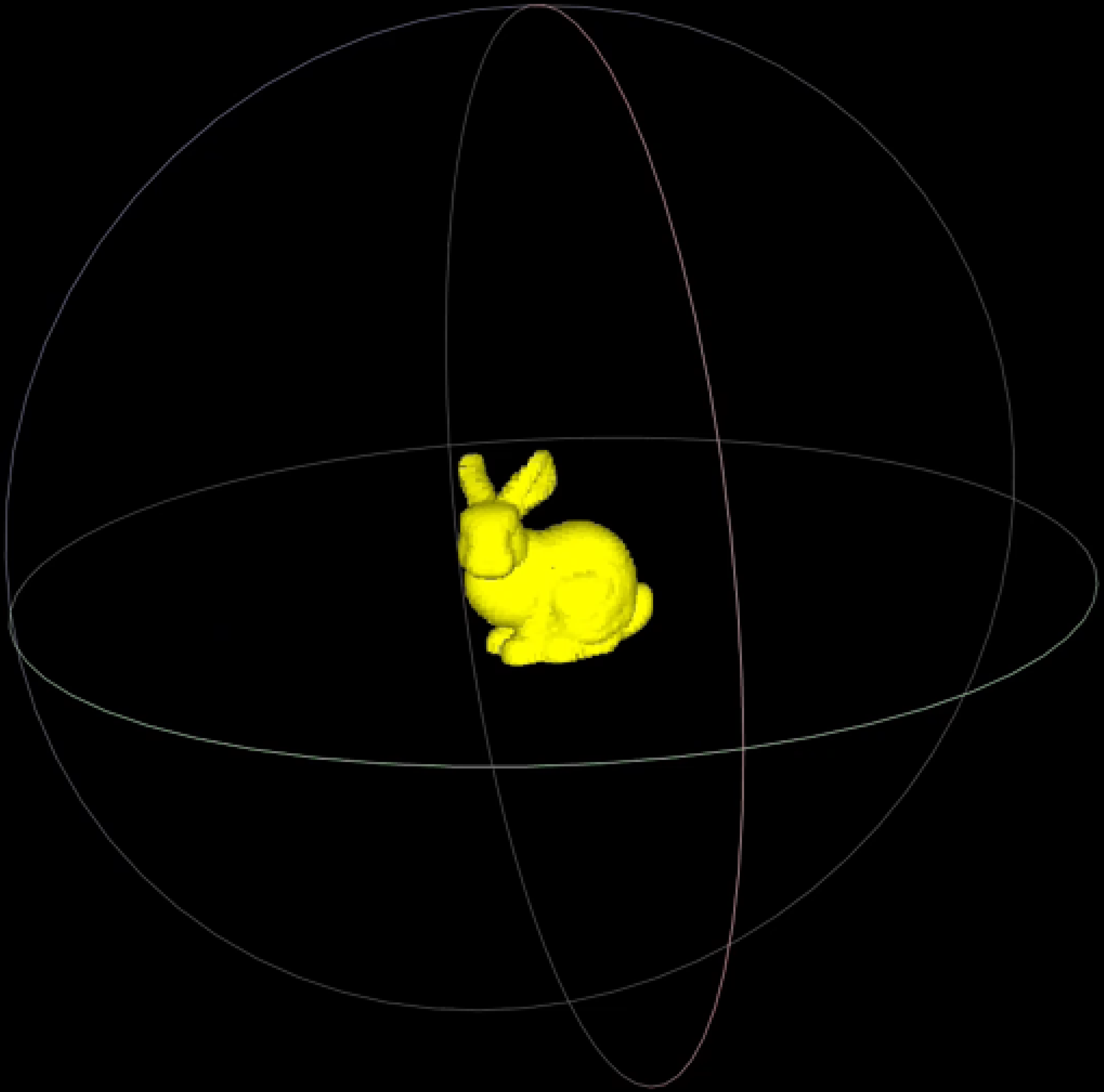


Semantic Segmentation

Extending Graph CNNs to Pointclouds

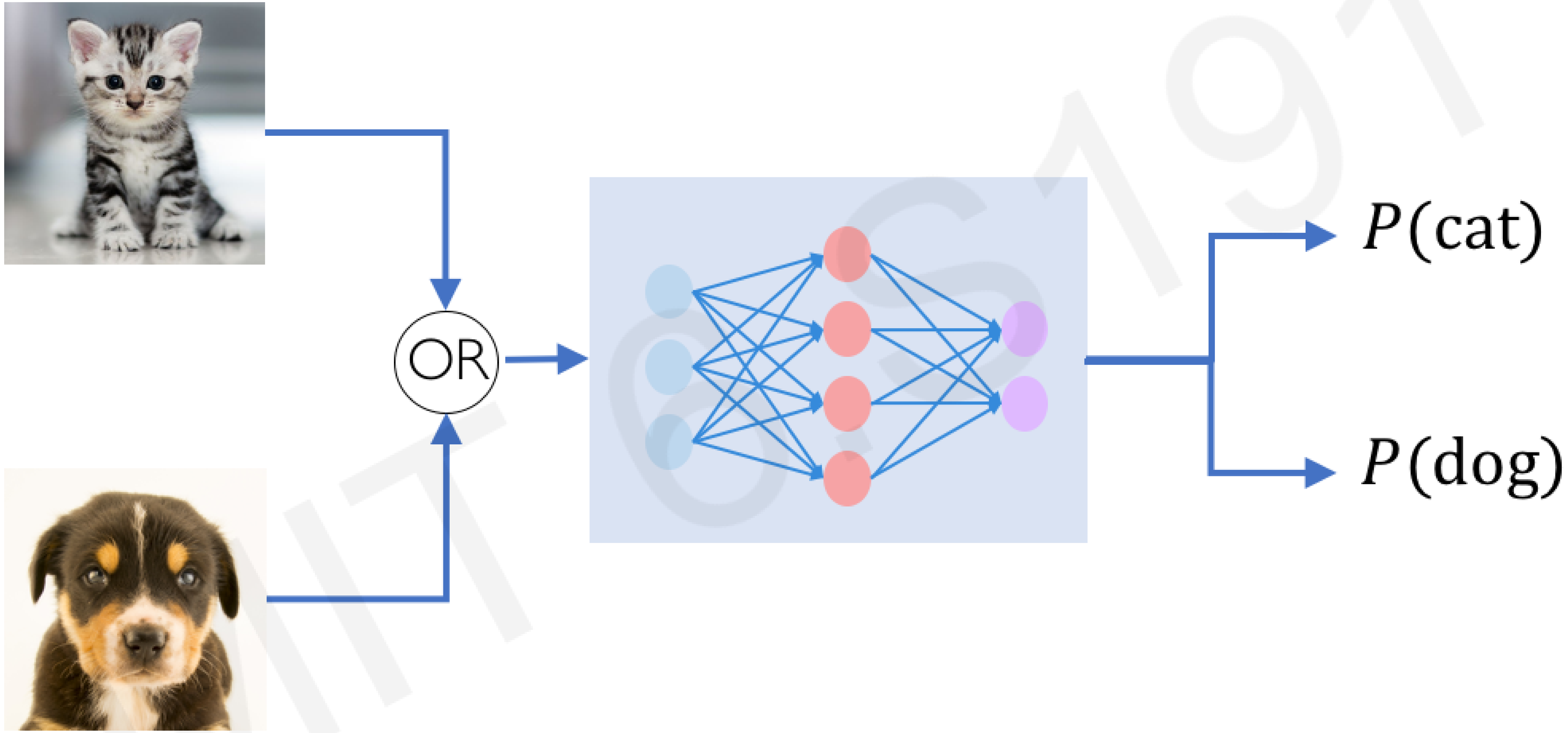
Capture local geometric features of point clouds while maintaining order invariance





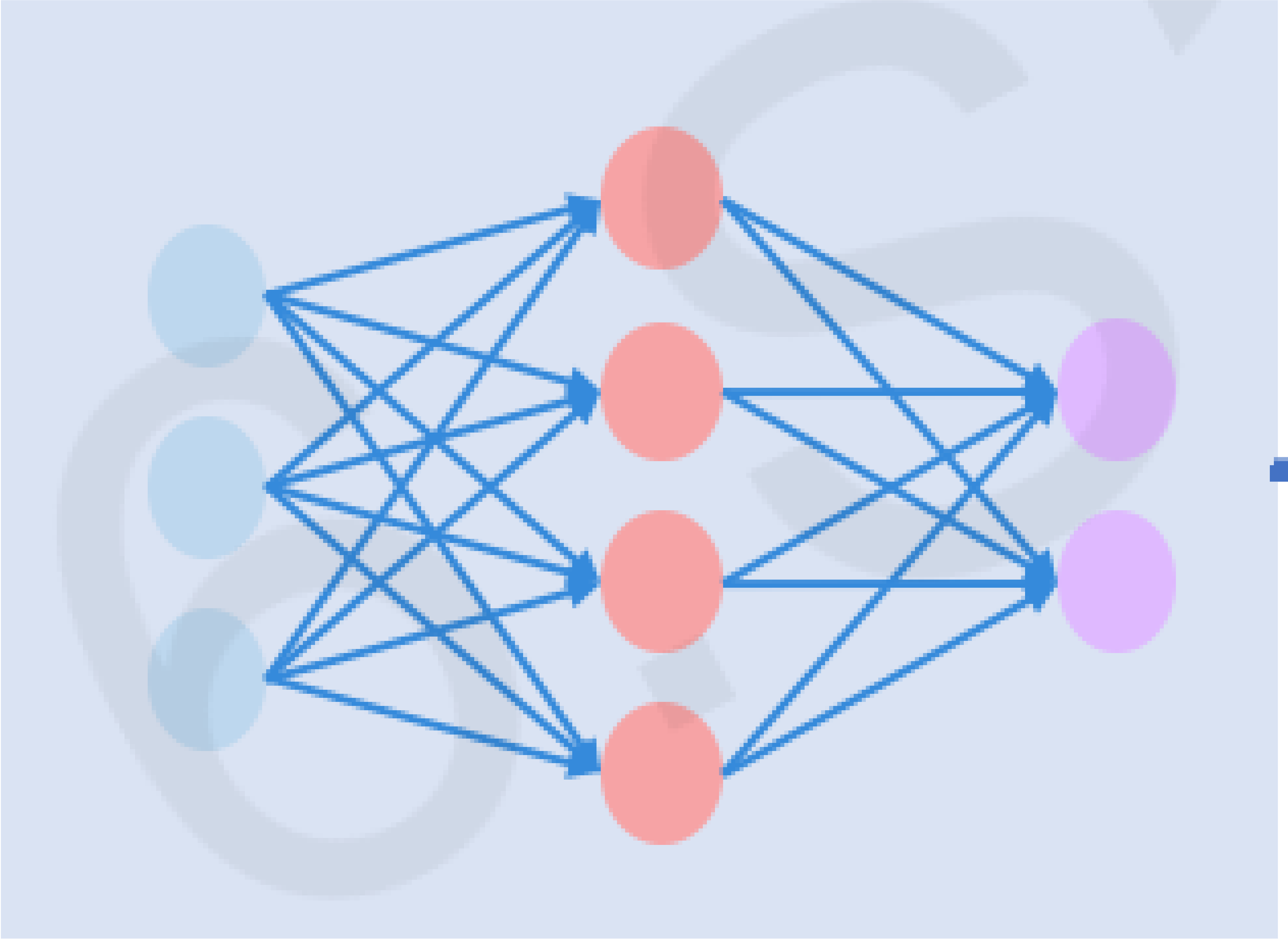
New Frontiers II:
Uncertainty Estimation & Bayesian Deep Learning

Why care about uncertainty?



Why care about uncertainty?

We need **uncertainty** metrics to assess the network's **confidence** in its predictions.



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

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

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

Bayesian Deep Learning for Uncertainty

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

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

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

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

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

Bayesian Deep Learning for Uncertainty

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

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

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

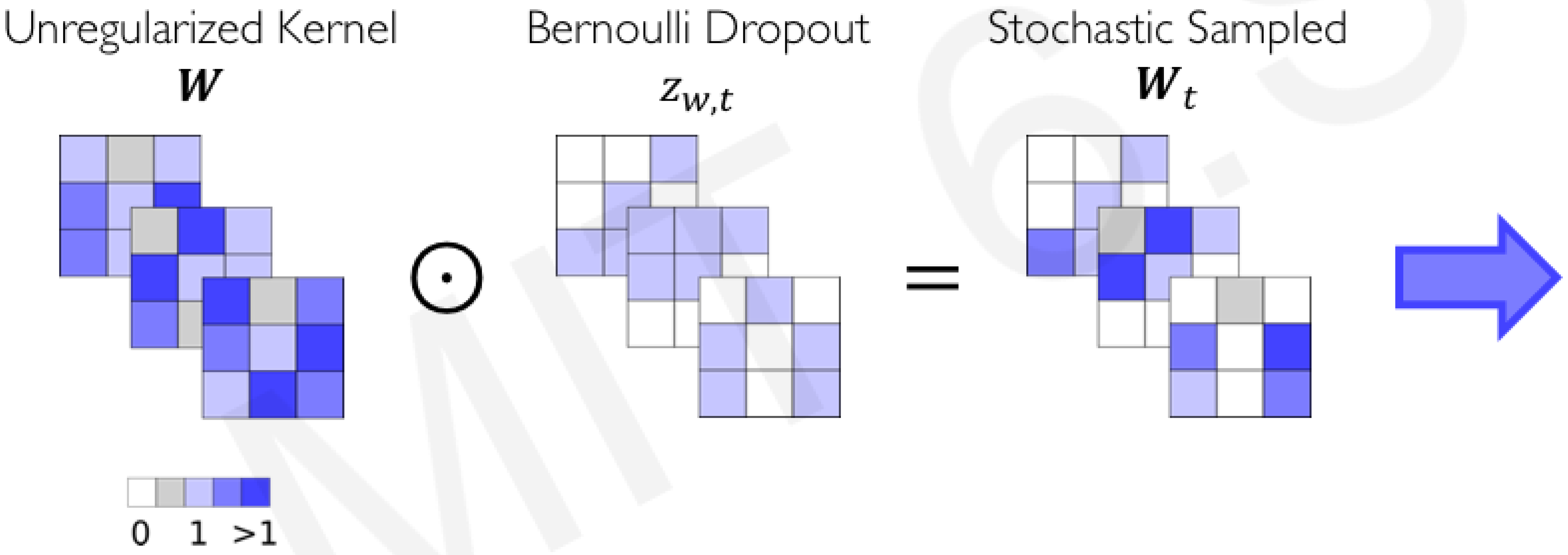
Bayesian neural networks aim to learn a posterior over weights, $P(\boldsymbol{W}|\boldsymbol{X},\boldsymbol{Y})$:

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

Dropout for Uncertainty

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

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

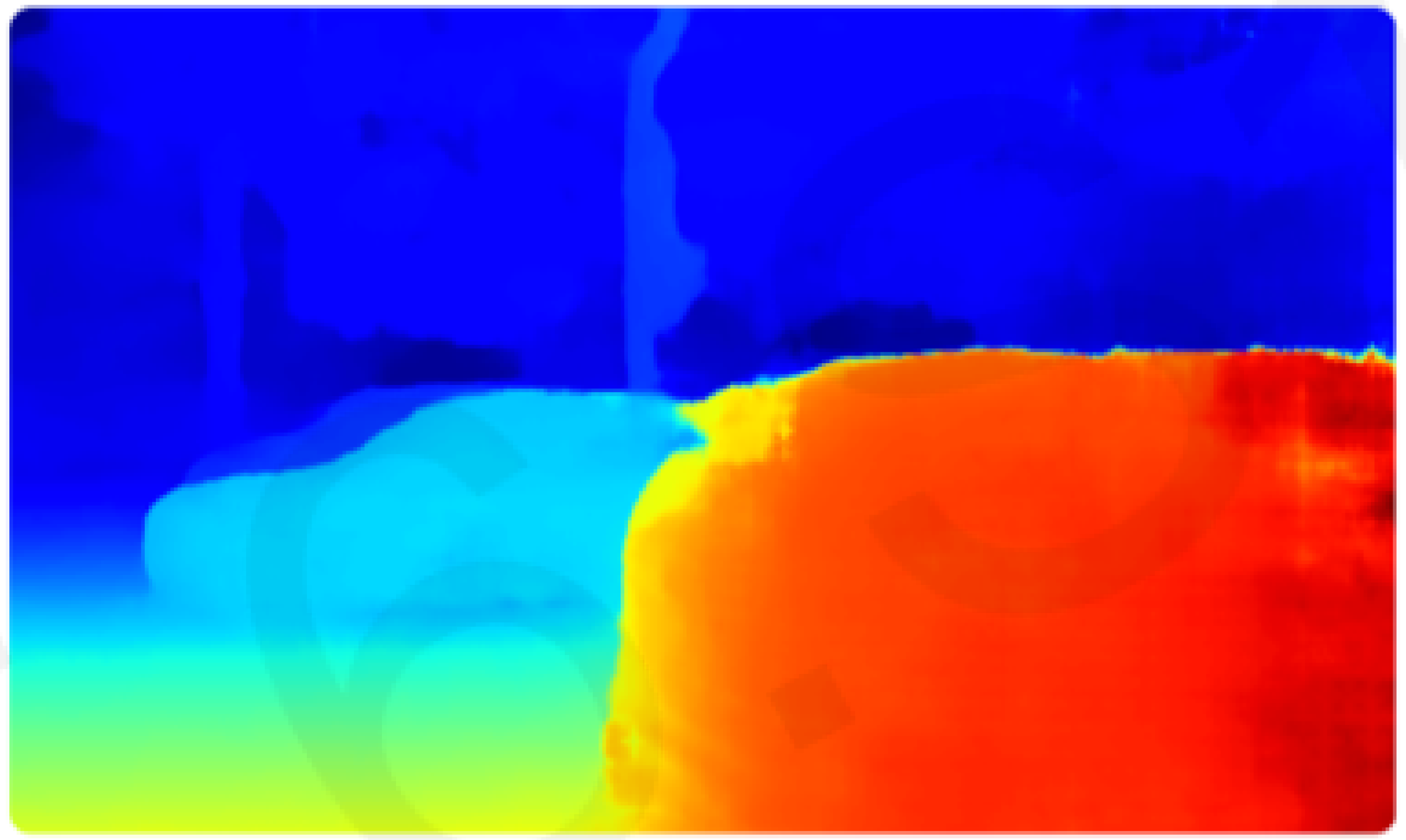


$$\mathbb{E}(\widehat{\boldsymbol{Y}}|\boldsymbol{X}) = \frac{1}{T}\sum_{t=1}^{T} f(\boldsymbol{X}|\boldsymbol{W}_t)$$

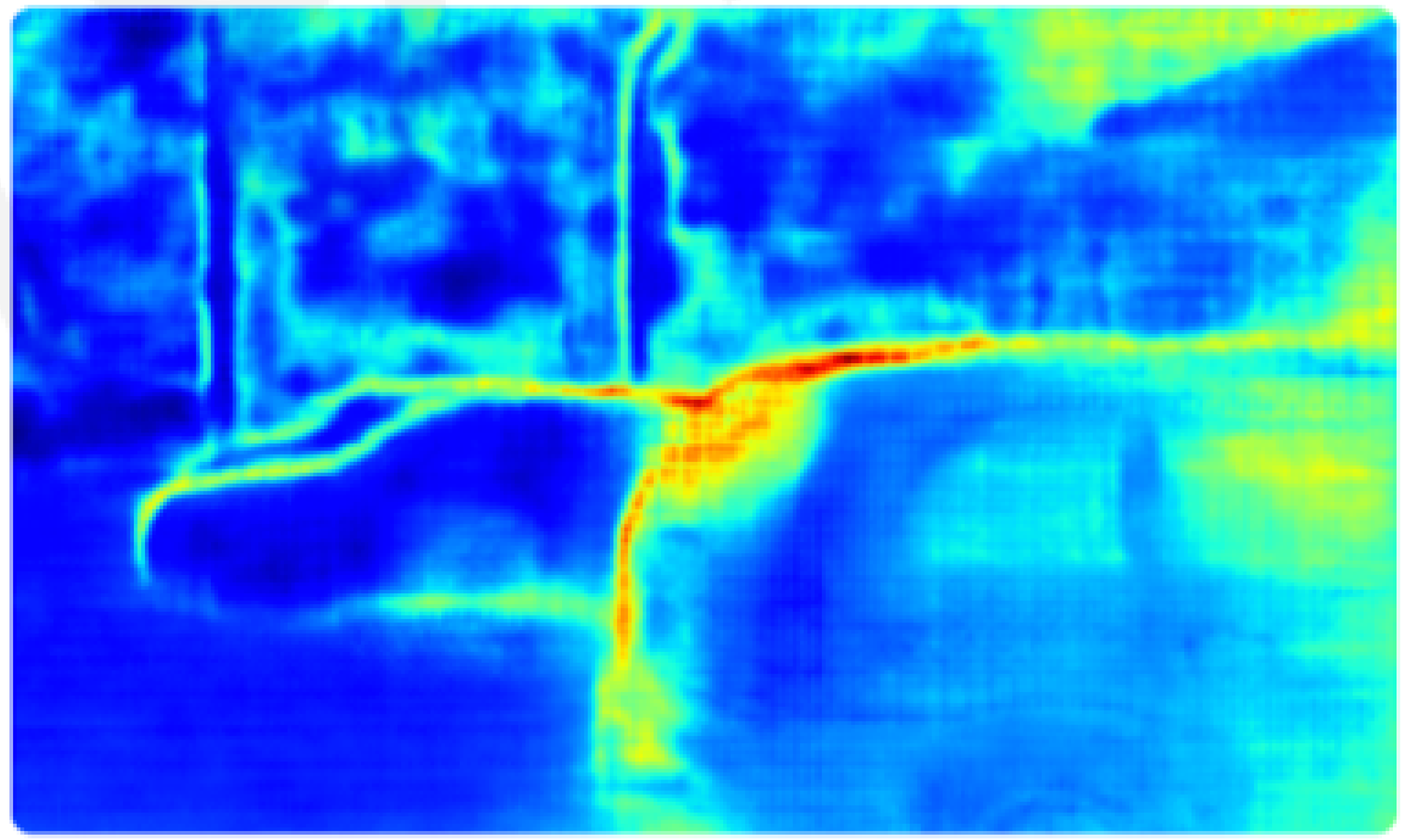
$$Var(\widehat{\boldsymbol{Y}}|\boldsymbol{X}) = \frac{1}{T}\sum_{t=1}^{T} f(\boldsymbol{X})^2 - \mathbb{E}(\widehat{\boldsymbol{Y}}|\boldsymbol{X})^2$$

Model Uncertainty Application

Input Image



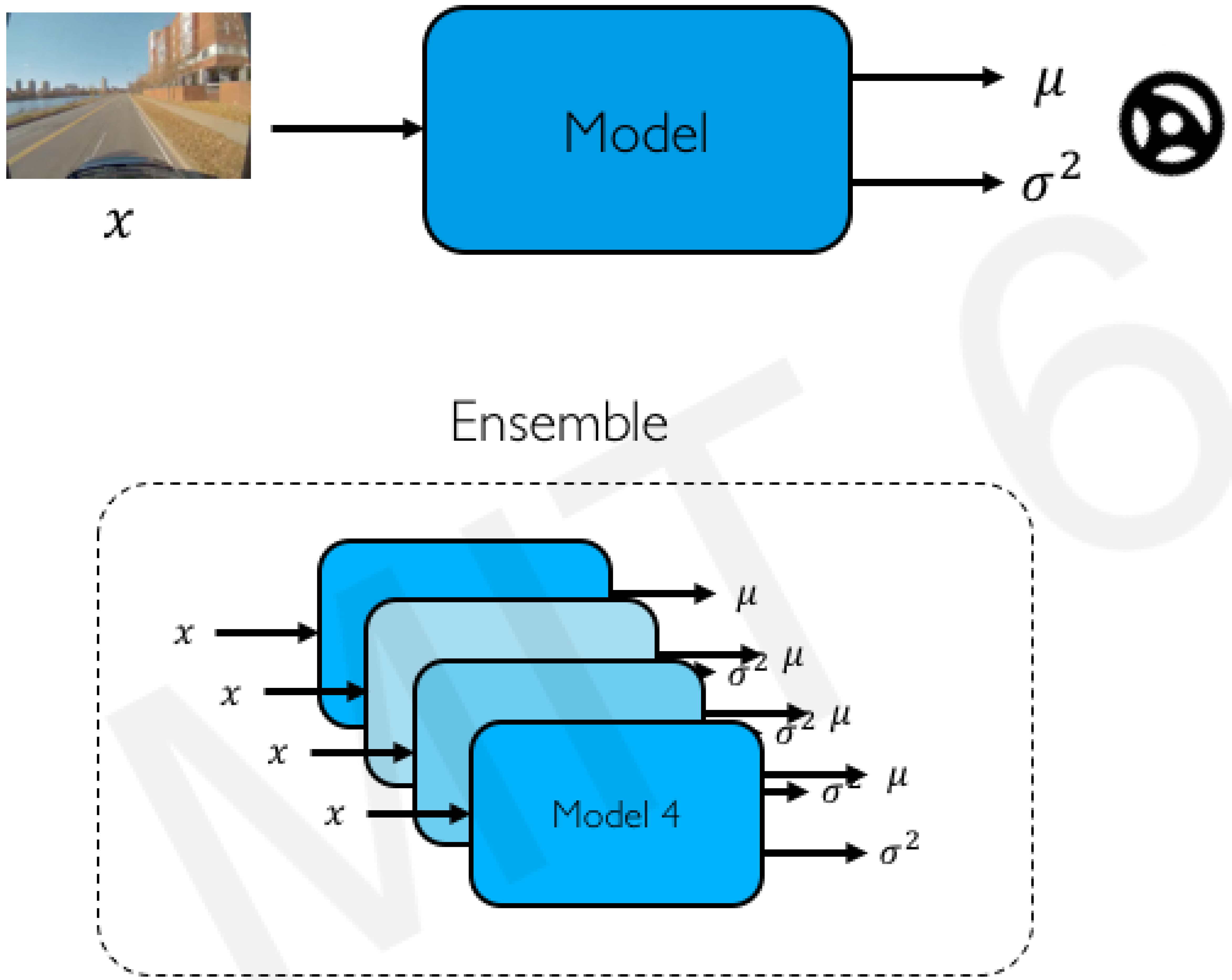
Predicted Depth

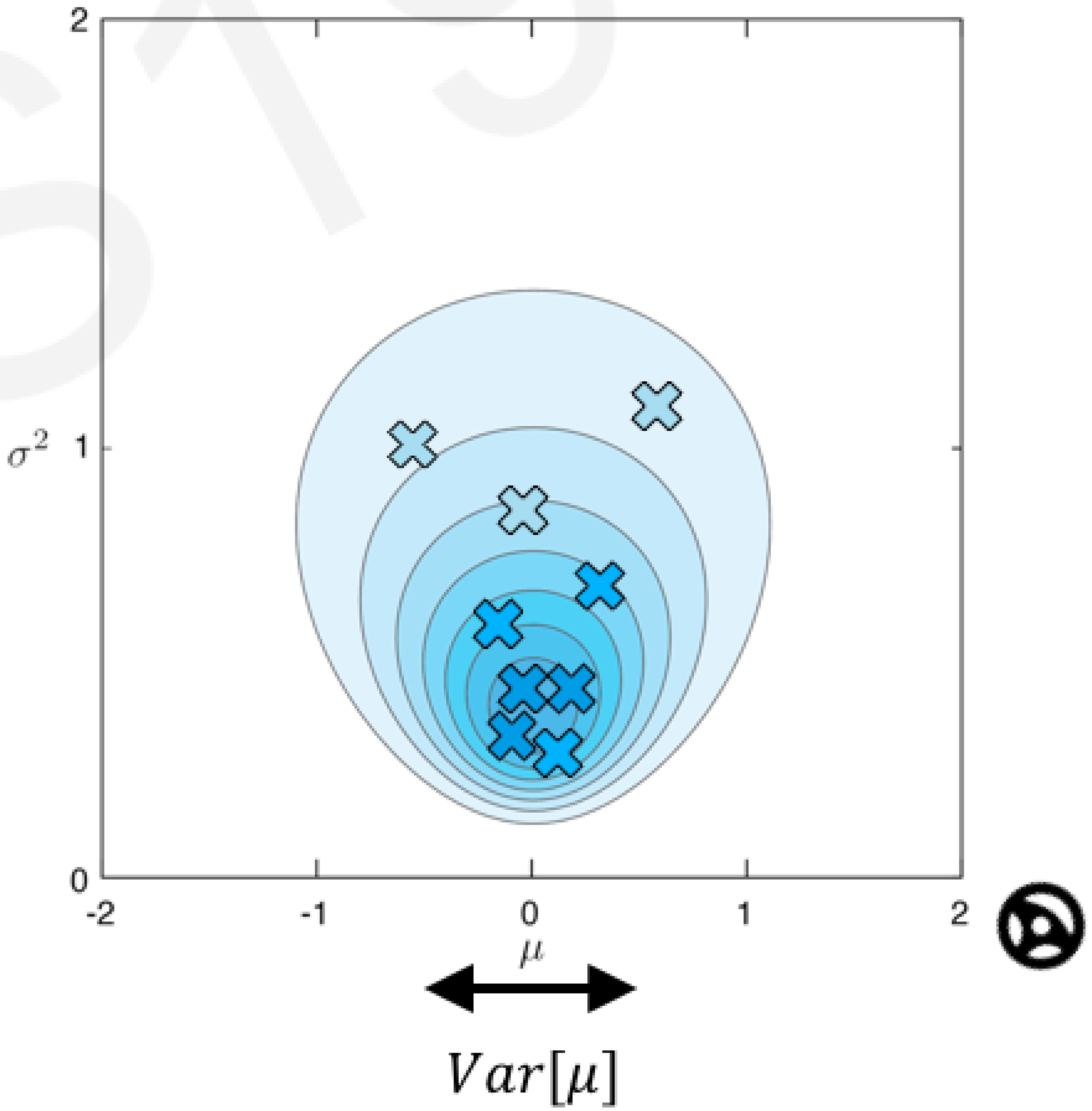


Model Uncertainty

Uncertainty Estimation via Ensembling

Model ensembling for estimating uncertainty

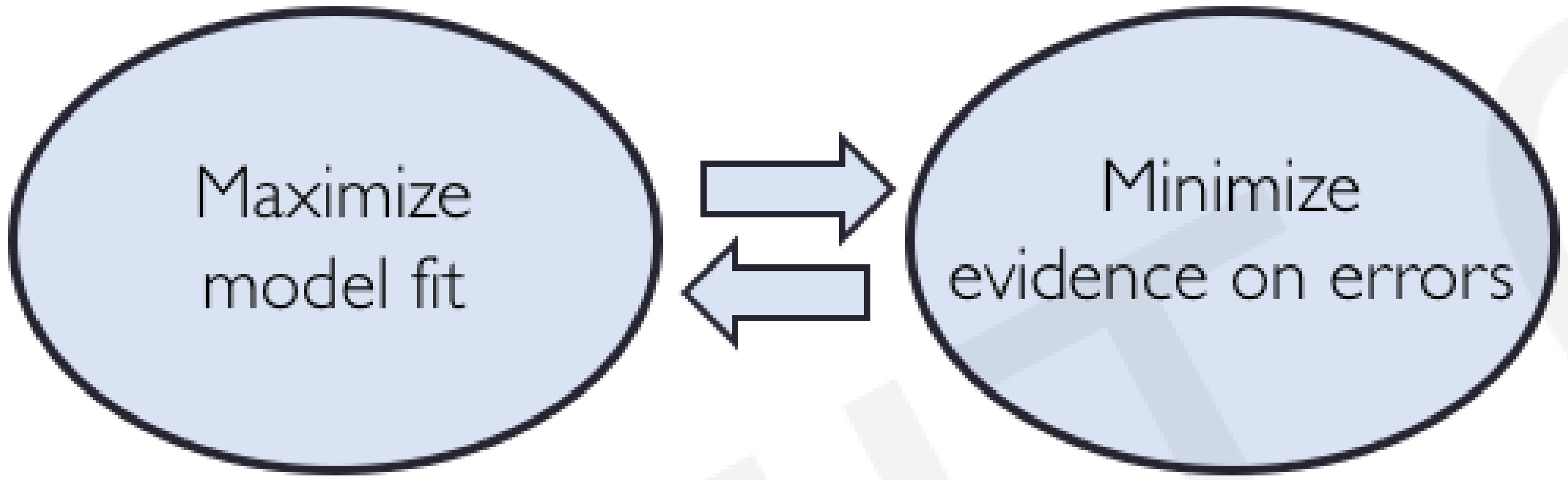


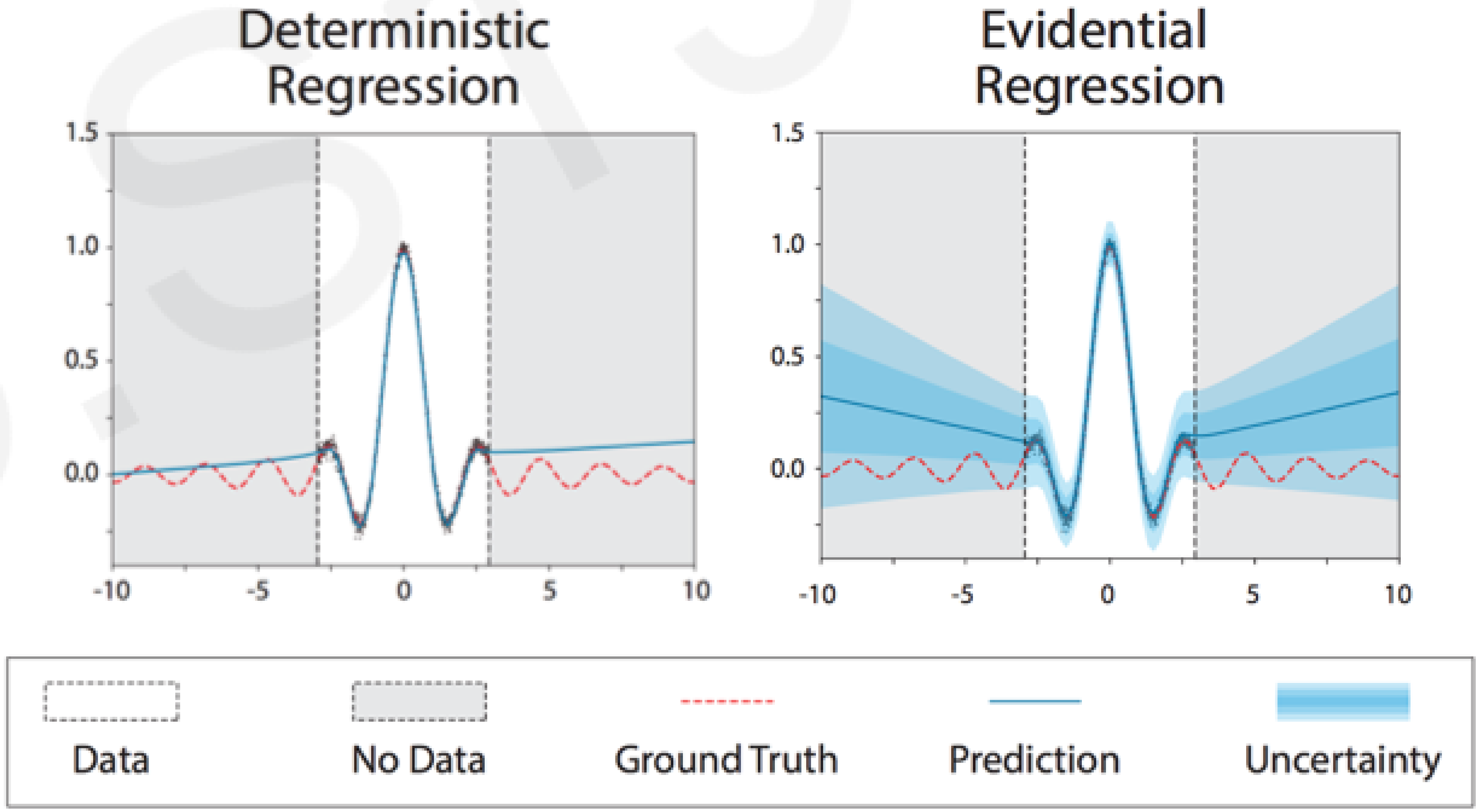


Evidential Deep Learning

Directly learn the underlying uncertainties using **evidential distributions**

Competing loss training:

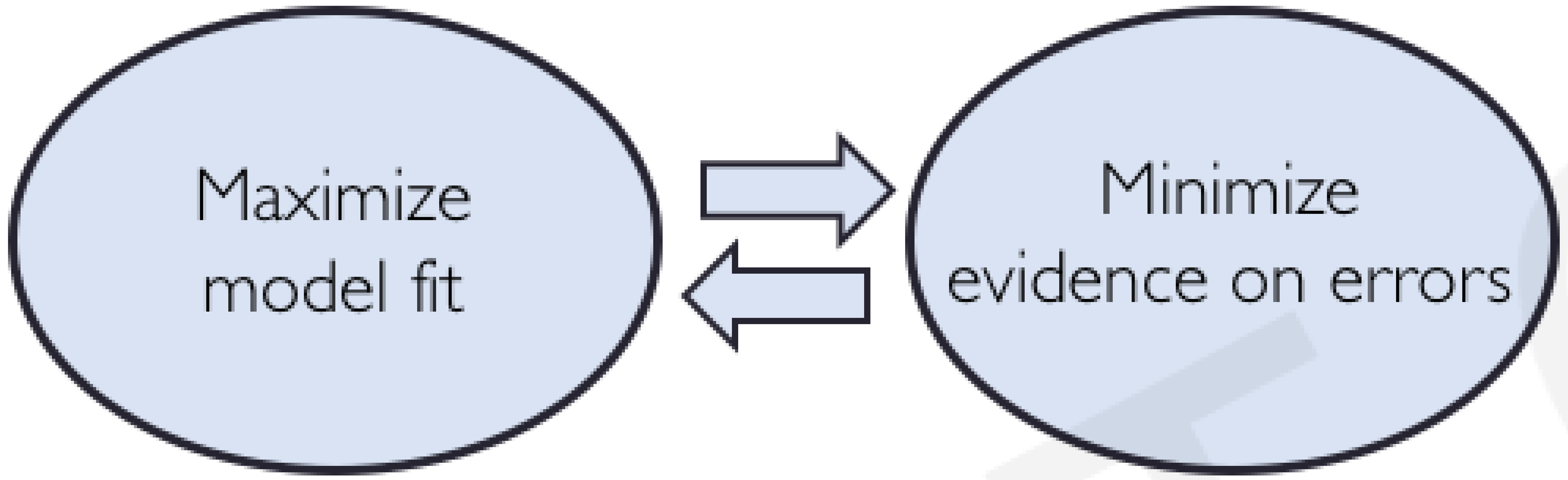


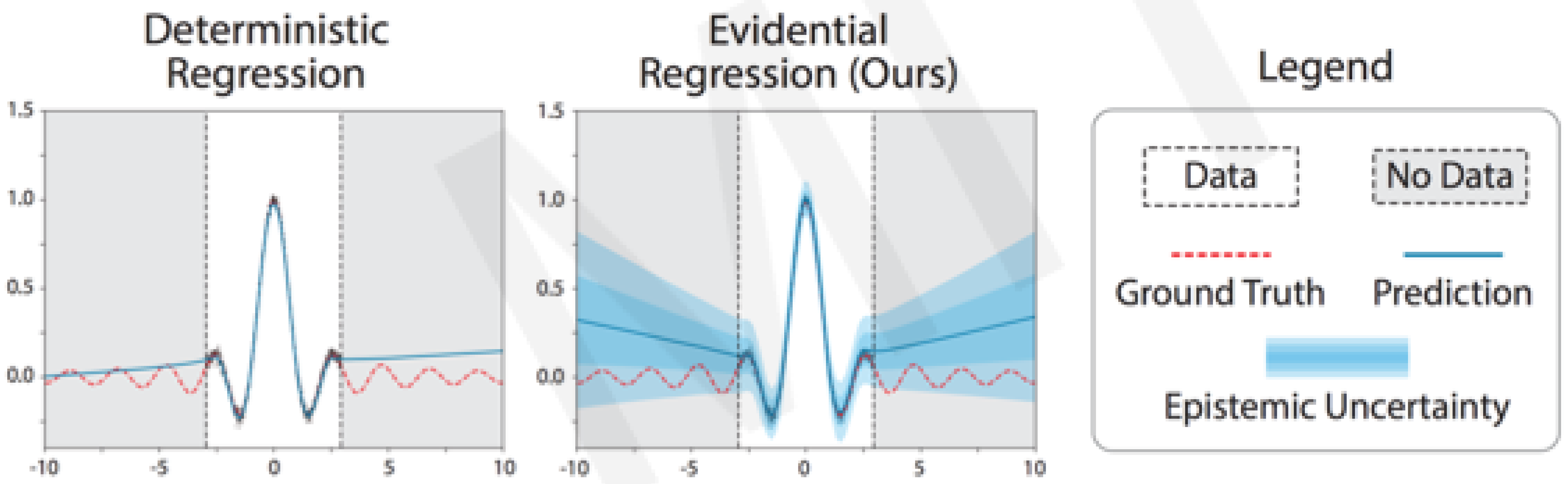


Evidential Deep Learning

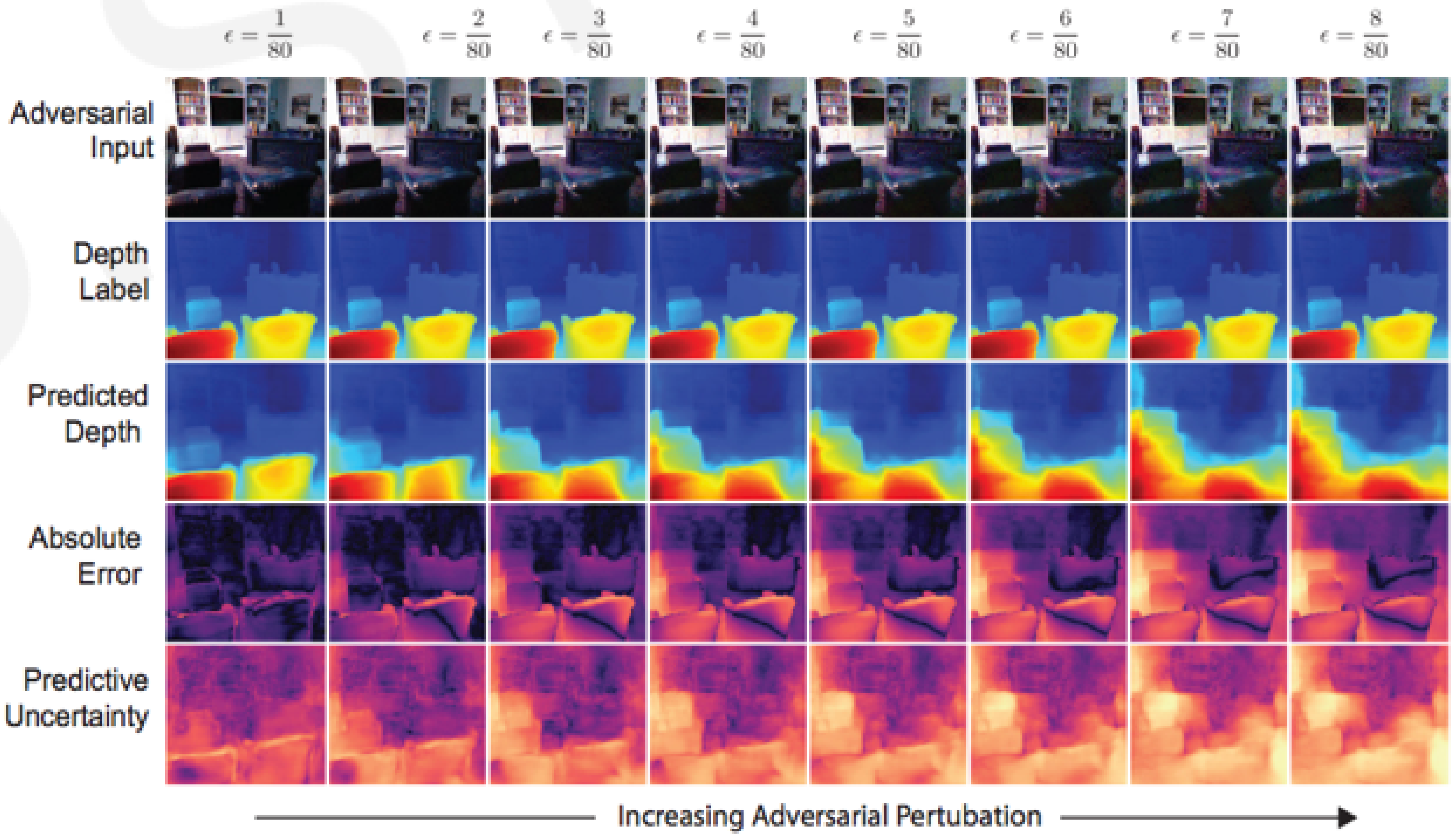
Directly learn the underlying uncertainties using **evidential distributions**

Competing loss training:



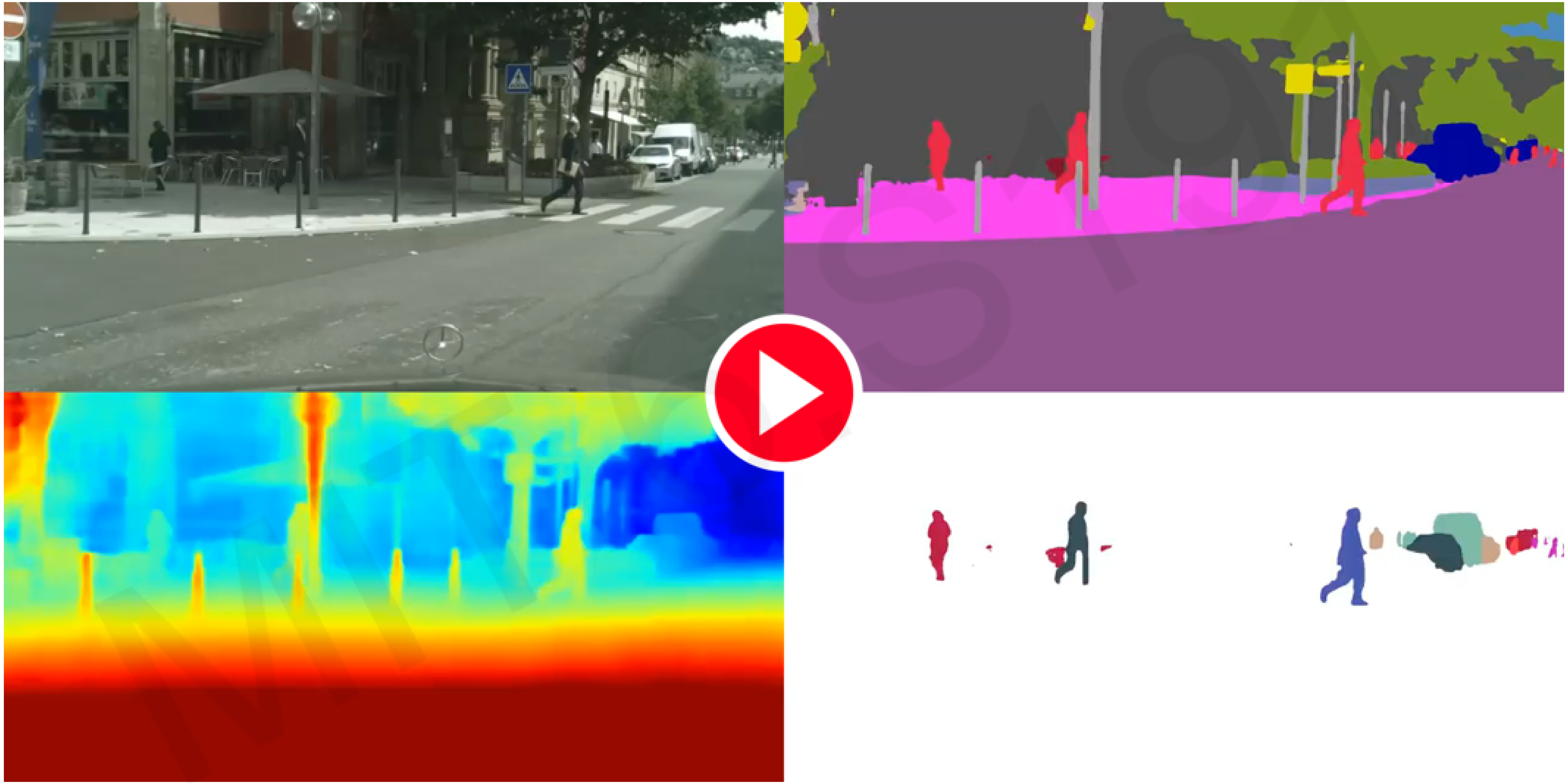


Robustness to adversarial perturbation



Multi-Task Learning Using Uncertainty

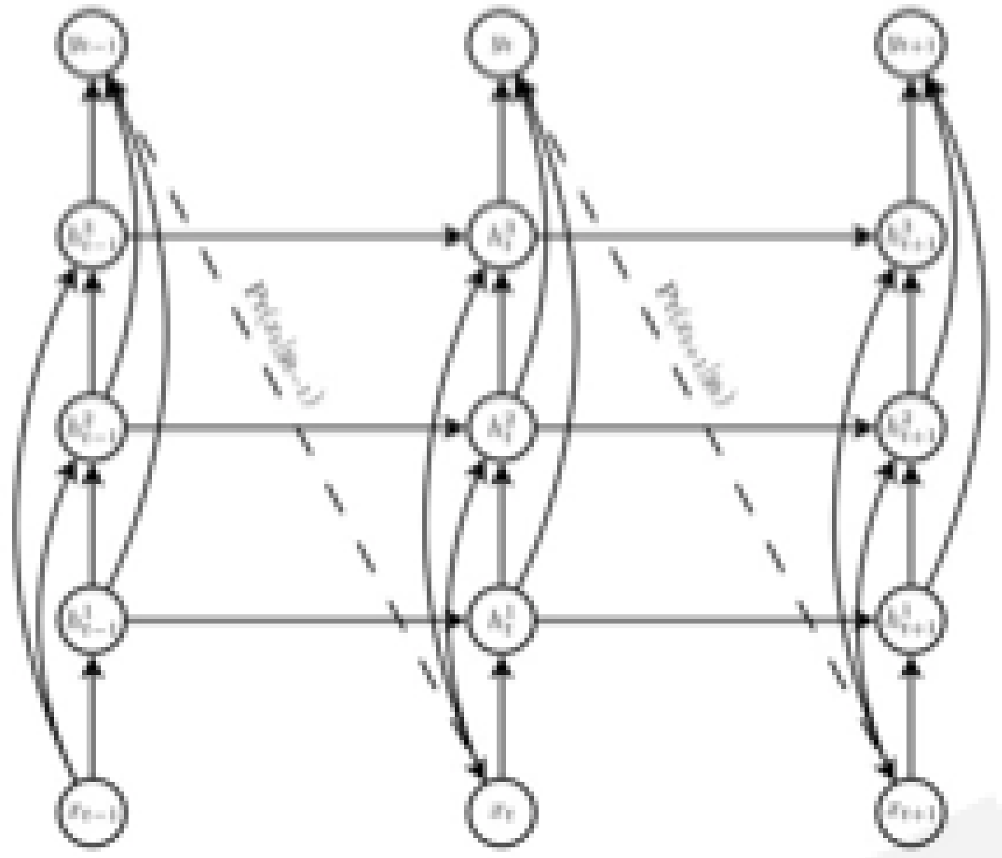
Multi-Task Learning Using Uncertainty



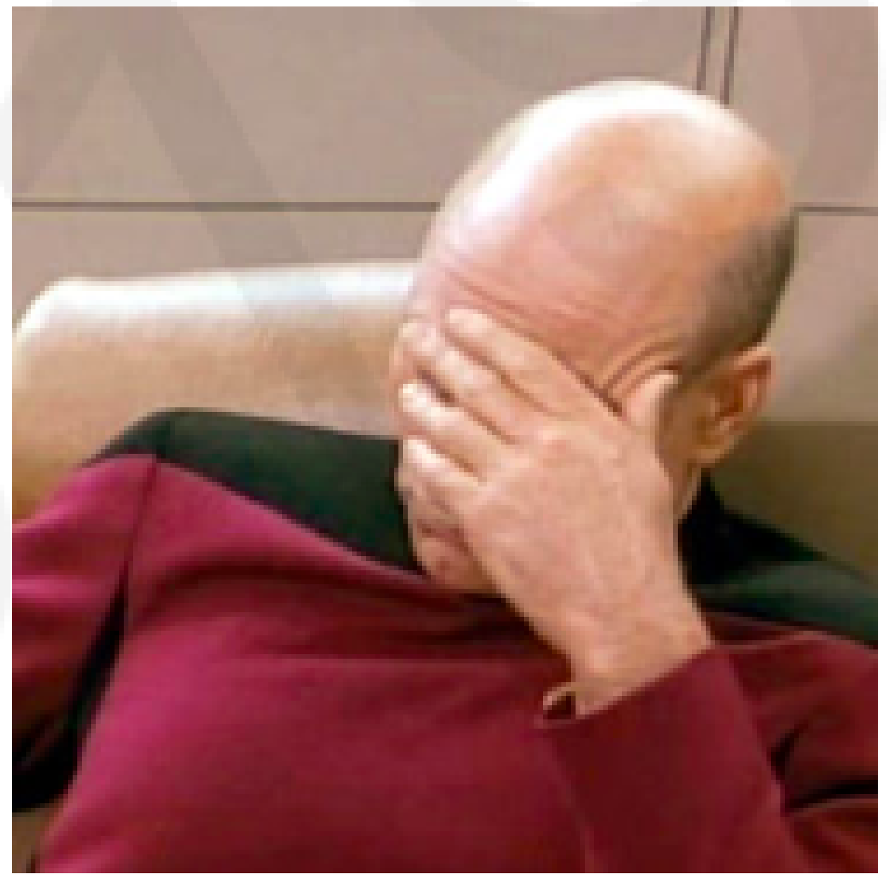
New Frontiers III:
Automated Machine Learning

Motivation: Automated Machine Learning

Standard deep neural networks are optimized for **a single task**



Complexity of models increases

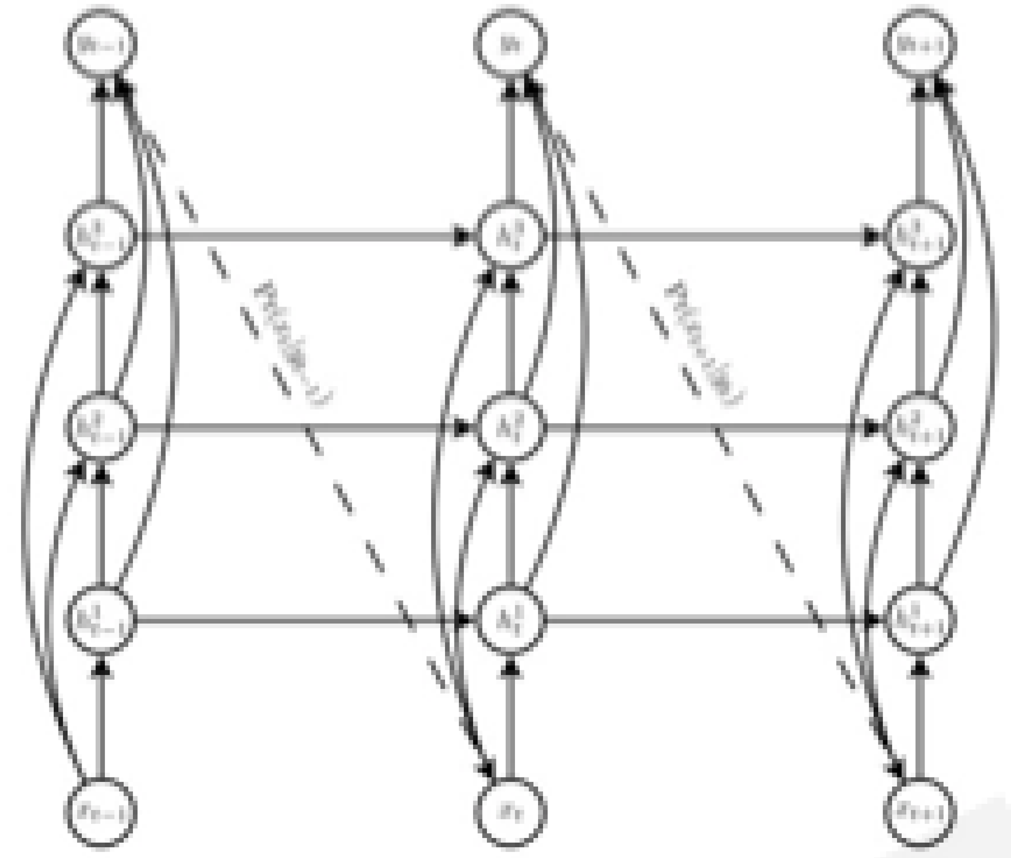


Greater need for specialized engineers

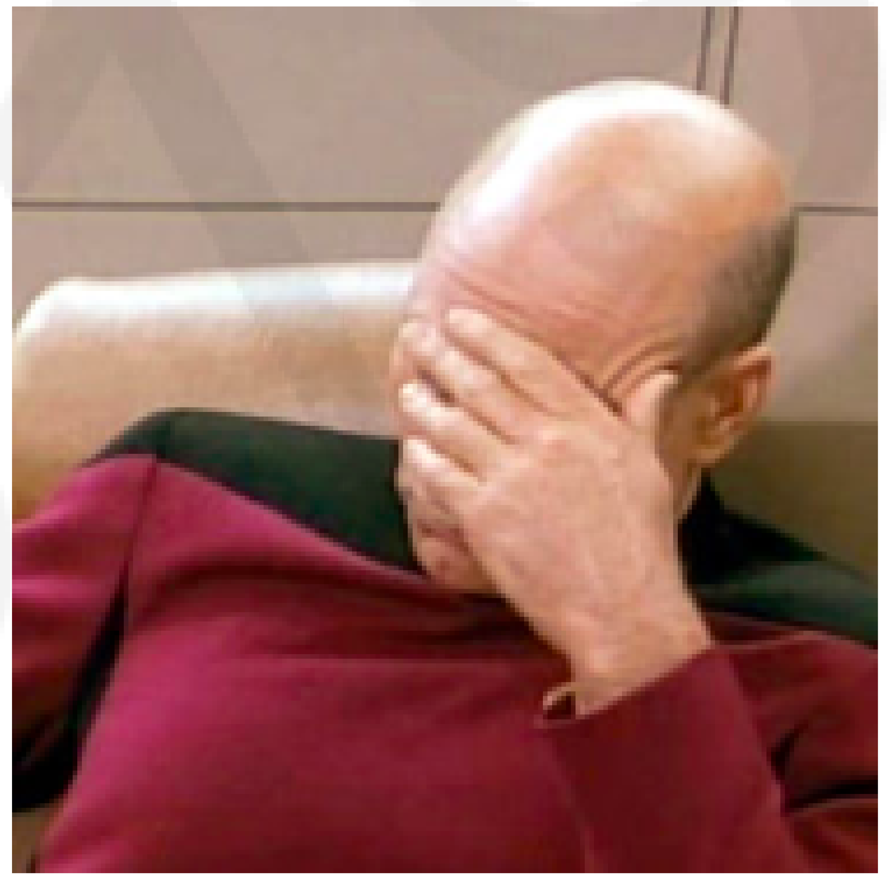
Often require **expert knowledge** to build an architecture for a given task

Motivation: Automated Machine Learning

Standard deep neural networks are optimized for **a single task**



Complexity of models increases



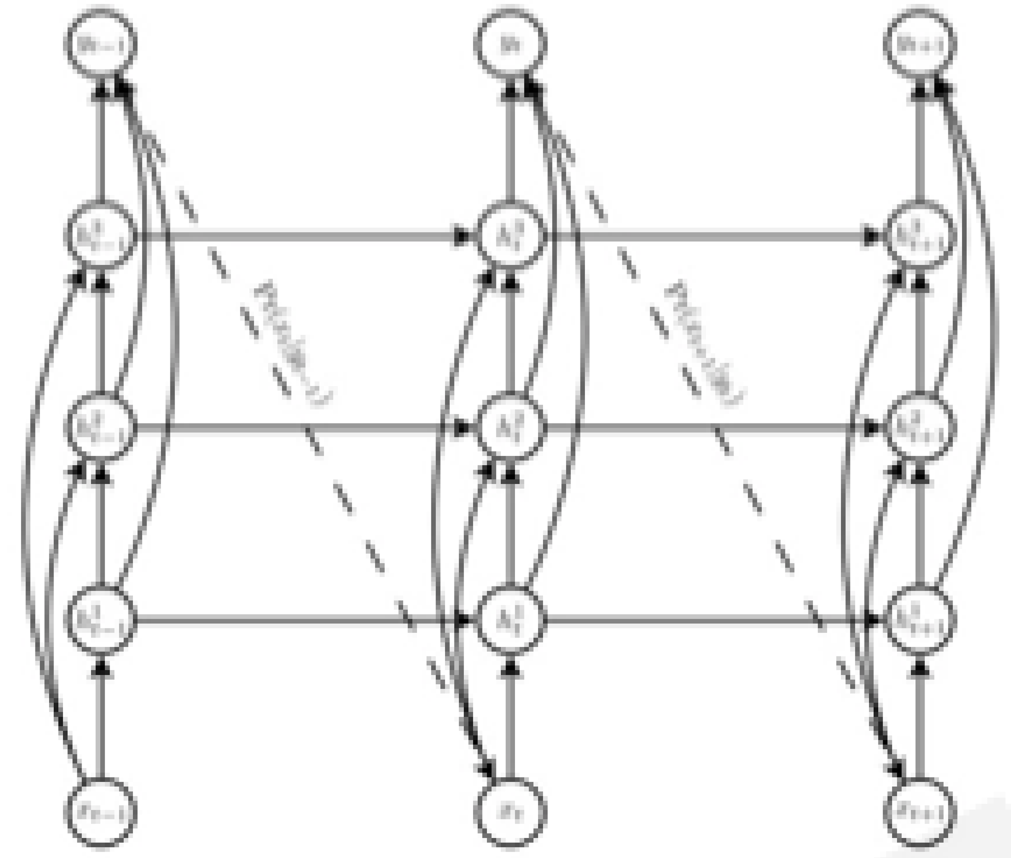
Greater need for specialized engineers

Often require **expert knowledge** to build an architecture for a given task

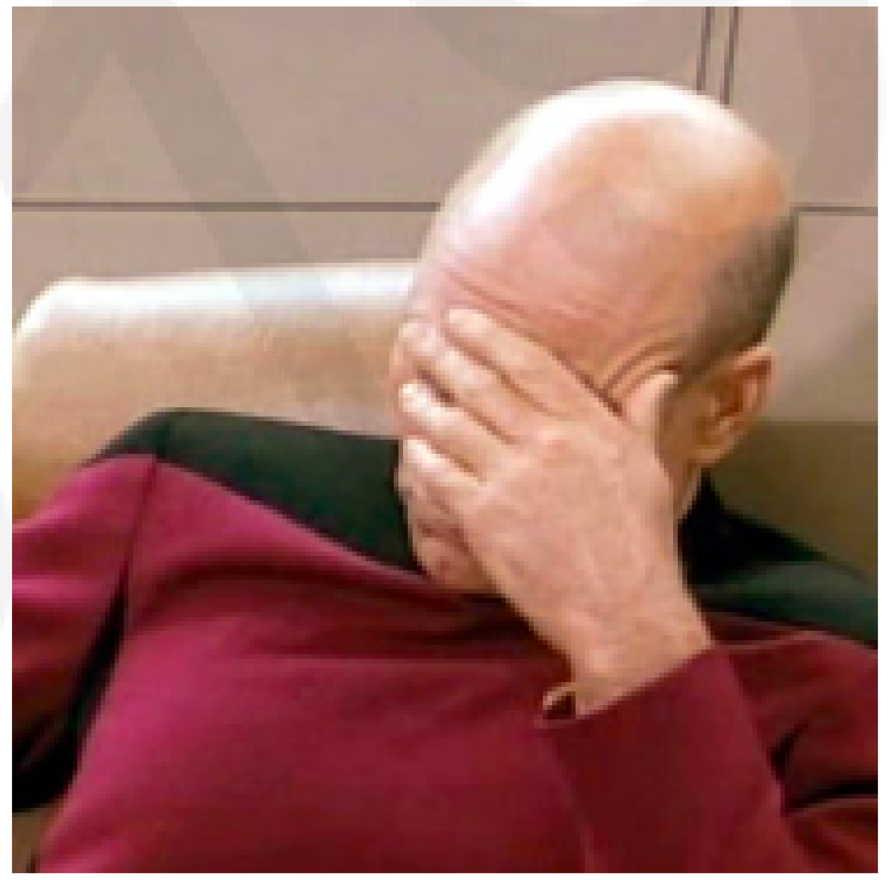
Build a learning algorithm that **learns which model** to use to solve a given problem

Motivation: Automated Machine Learning

Standard deep neural networks are optimized for **a single task**



Complexity of models increases



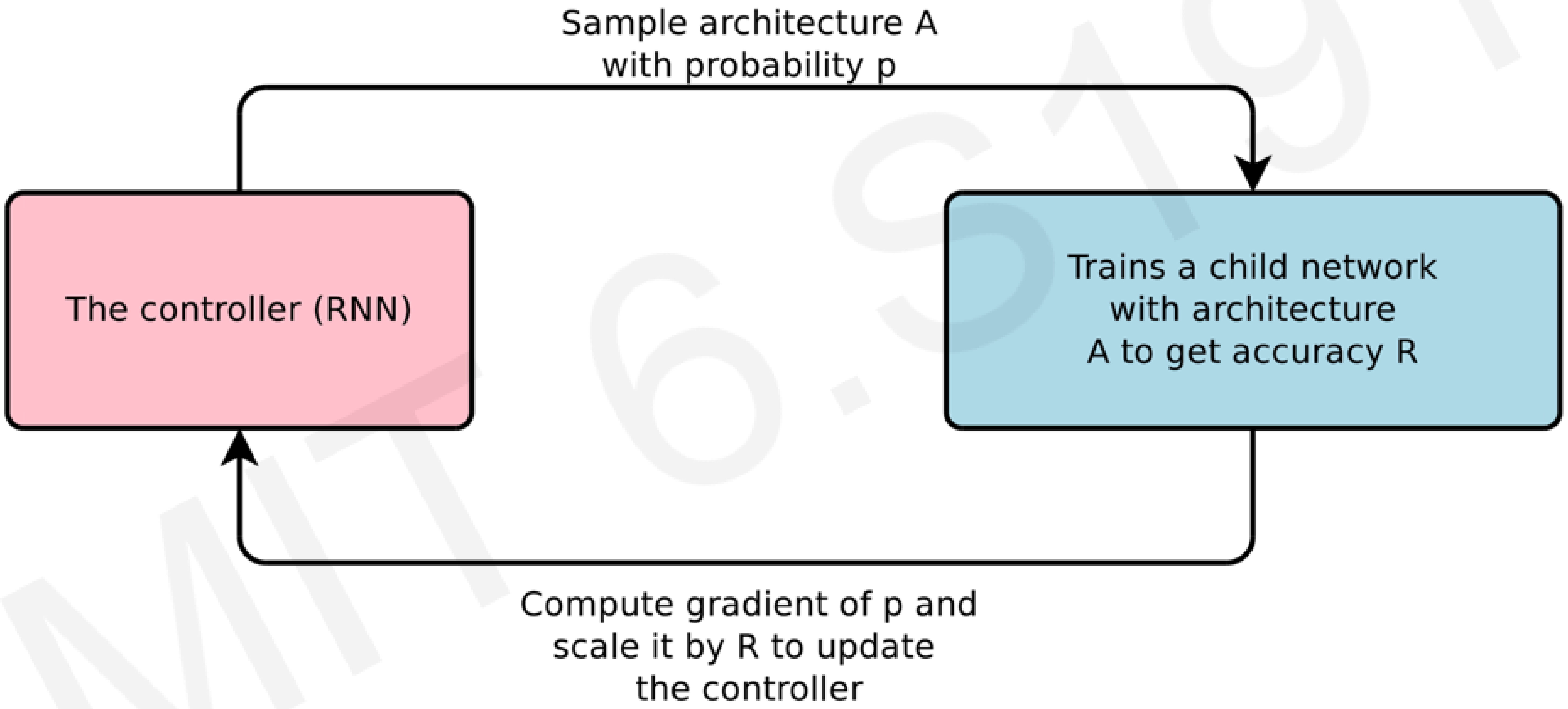
Greater need for specialized engineers

Often require **expert knowledge** to build an architecture for a given task

Build a learning algorithm that **learns which model** to use to solve a given problem

AutoML

Automated Machine Learning (AutoML)



AutoML: Model Controller

At each step, the model samples a brand new network

AutoML: The Child Network

Training Data → Sampled network from RNN → Prediction

Compute final accuracy on this dataset.

Update RNN controller based on the accuracy of the child network after training.

AutoML on the Cloud

AutoML Vision BETA

Start with as little as a few dozen photographic samples, and Cloud AutoML will do the rest.

AutoML Natural Language BETA

Automatically predict text categories through either single or multi-label classification.

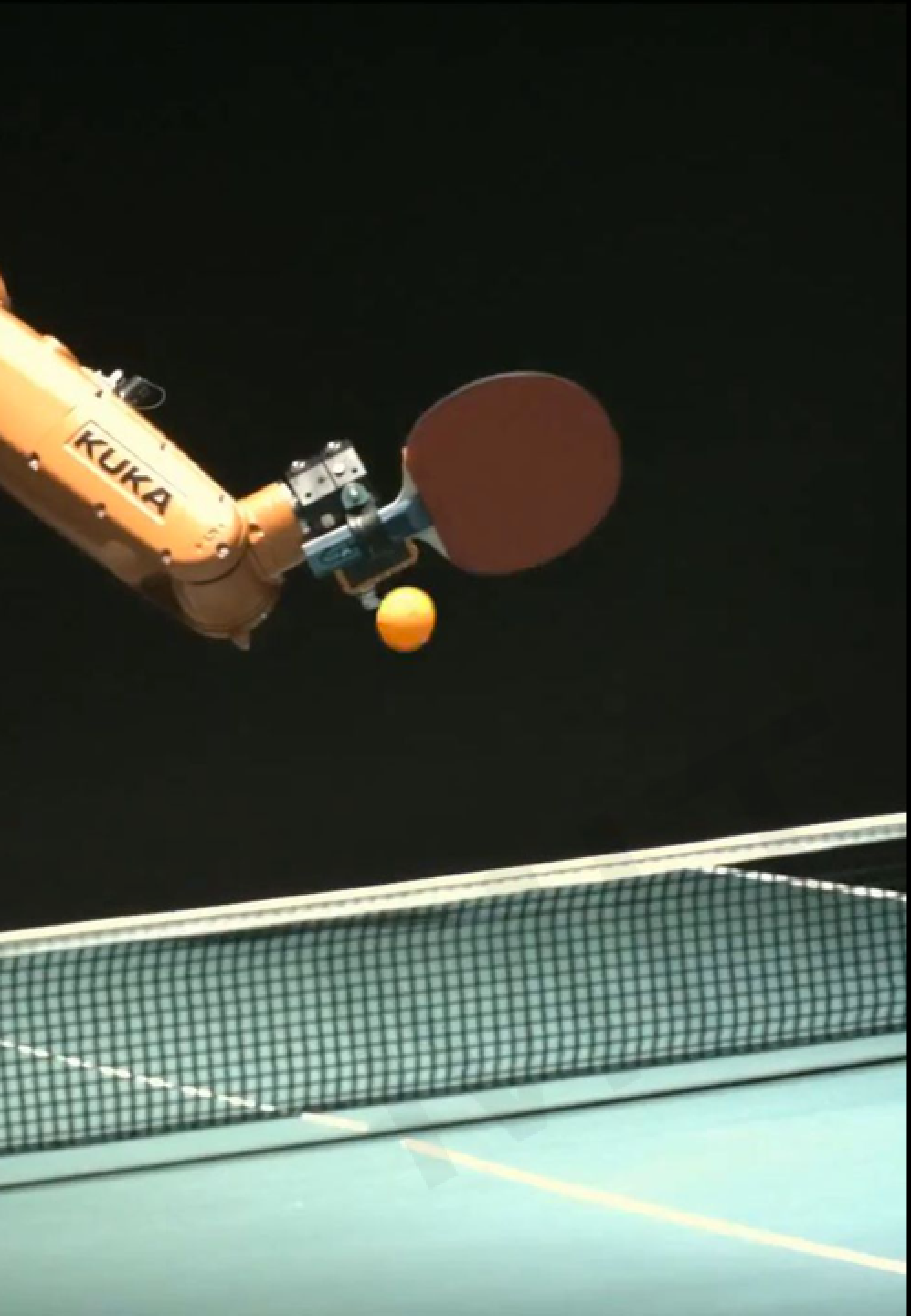
AutoML Translation BETA

Upload translated language pairs to train your own custom model.

AutoML Spawns a Powerful Idea

- Design an AI algorithm that can build new models capable of solving a task
- Reduces the need for experienced engineers to design the networks
- Makes deep learning more accessible to the public

Connections and distinctions between artificial and human intelligence



6.S191: Introduction to Deep Learning

Lab 3: Reinforcement Learning

Link to download labs:
http://introtodeeplearning.com#schedule

1. Open the lab in Google Colab
2. Start executing code blocks and filling in the #TODOs
3. Need help? Find a TA or come to the front!!