

# UVA CS 4774: Machine Learning



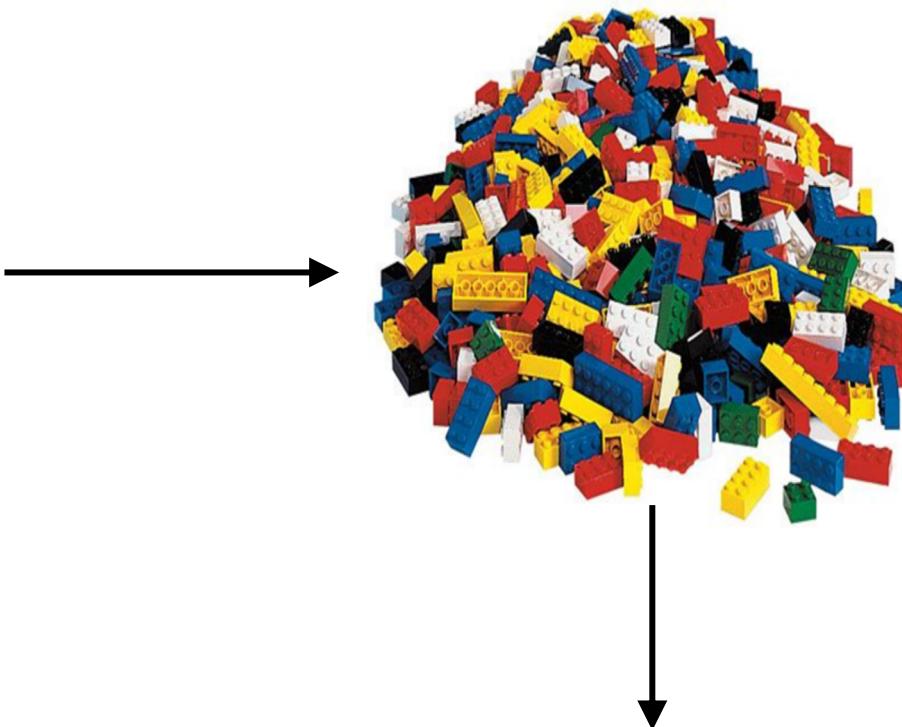
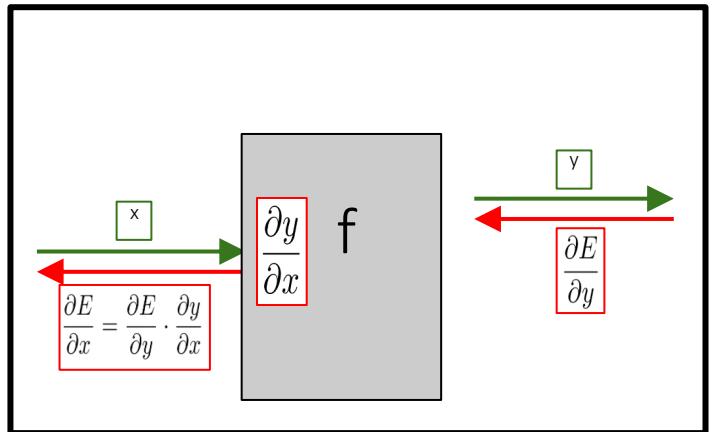
## S3: Lecture 19: Recent Deep Neural Networks: A Quick Overview

Dr. Yanjun Qi

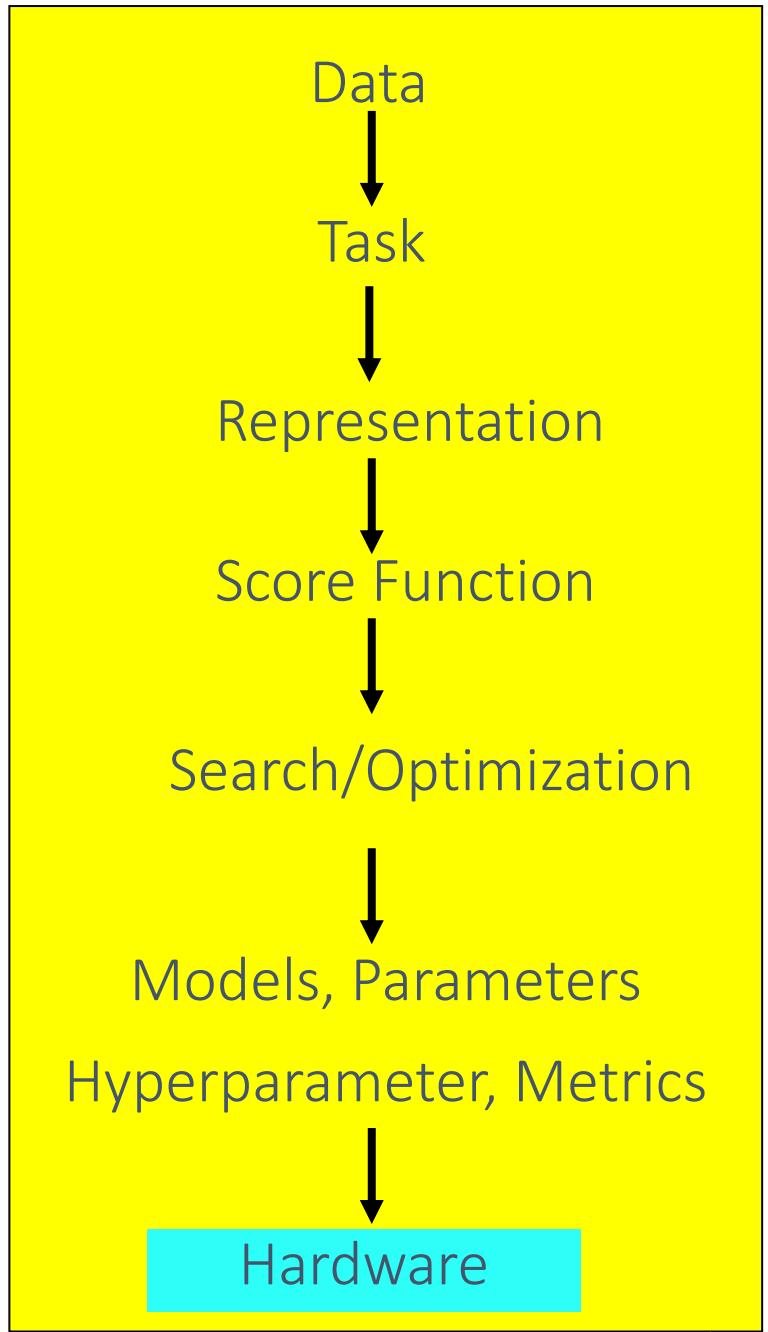
University of Virginia  
Department of Computer Science

Module I

# Recap: Building Deep Neural Nets



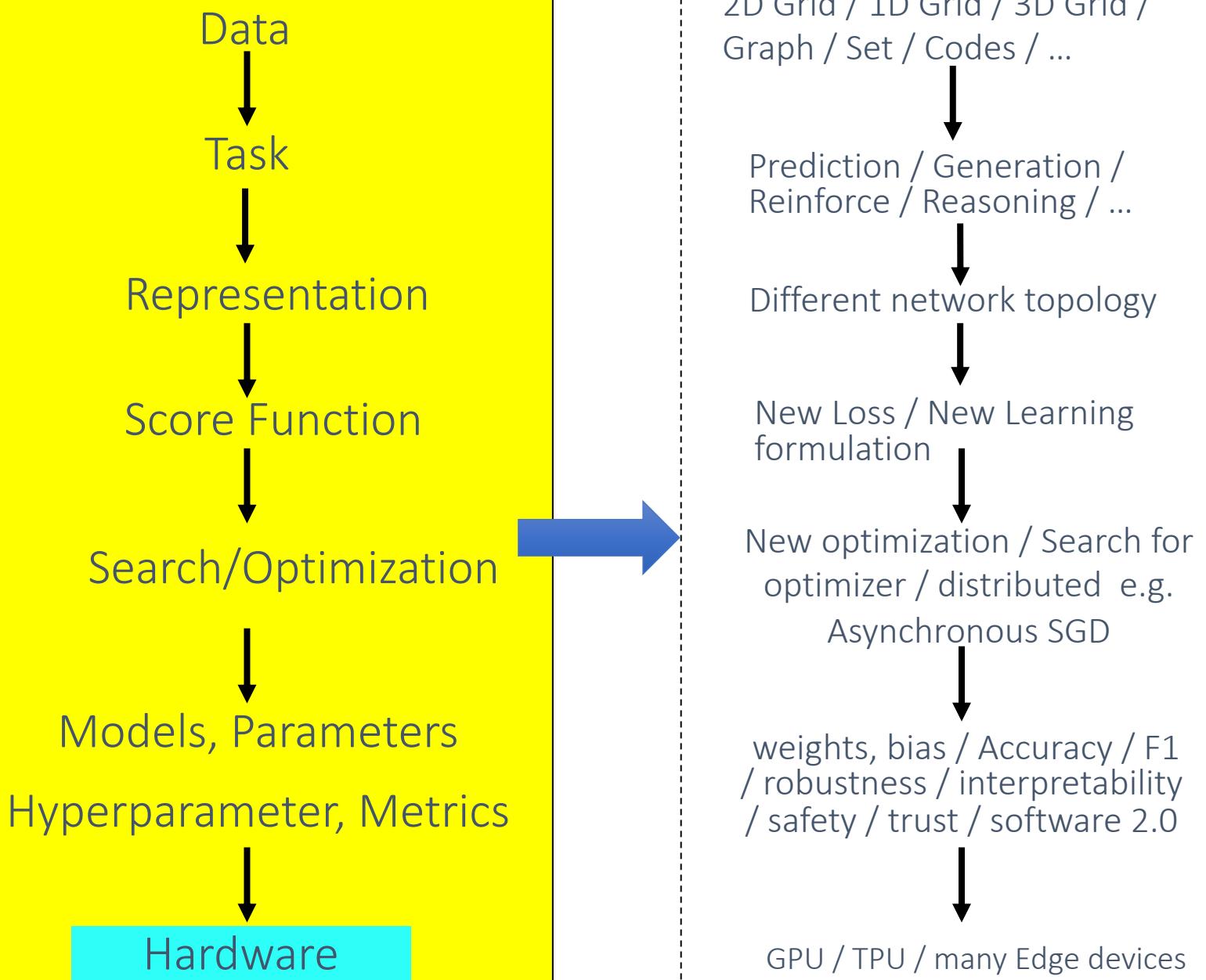
# Deep Learning in a Nutshell



Disclaimer: it is quite hard to make important topics of machine learning fit on a one-semester schedule.

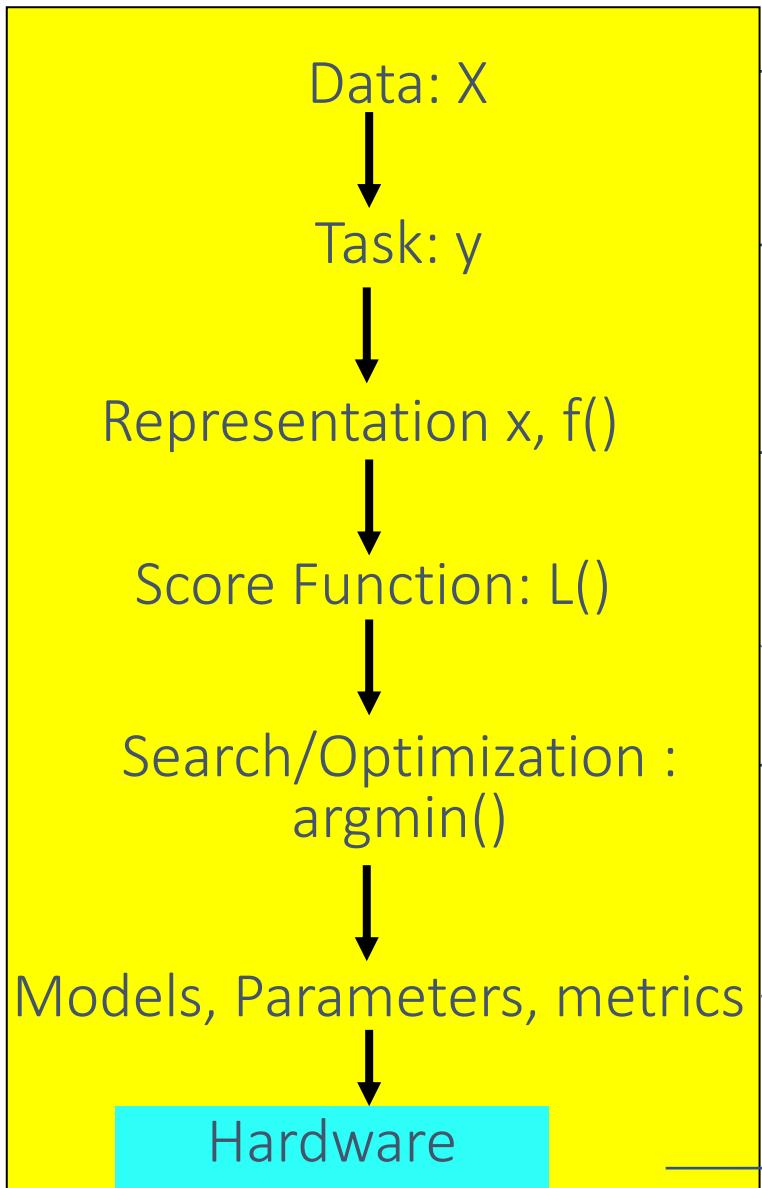
Disclaimer: it is quite hard to make important topics of deep learning fit on a one-session schedule.

We aim to make the content reasonably digestible in an introductory manner. We try to focus on a modularity view by introducing important variables to digest machine learning into chunks regarding data/ representation / loss-functions / optimizations / model characteristics. That said, our goals here are to highlight the most foundational design choices in machine learning about algorithm designs, workflows, what to learn and how to learn it, and to expose the trade-offs in those choices. We think this teaching style provides students with context concerning those choices and helps them build a much deeper understanding.



# Selected Trends

<https://qdata.github.io/deep2Read/>



- 1. DNN on graphs / trees / sets
- 2. NTM 4program induction

- 1. Deep Generative models/ DeepFake
- 2. Deep reinforcement learning
- 3. Few-shots / Meta learning / AGI?

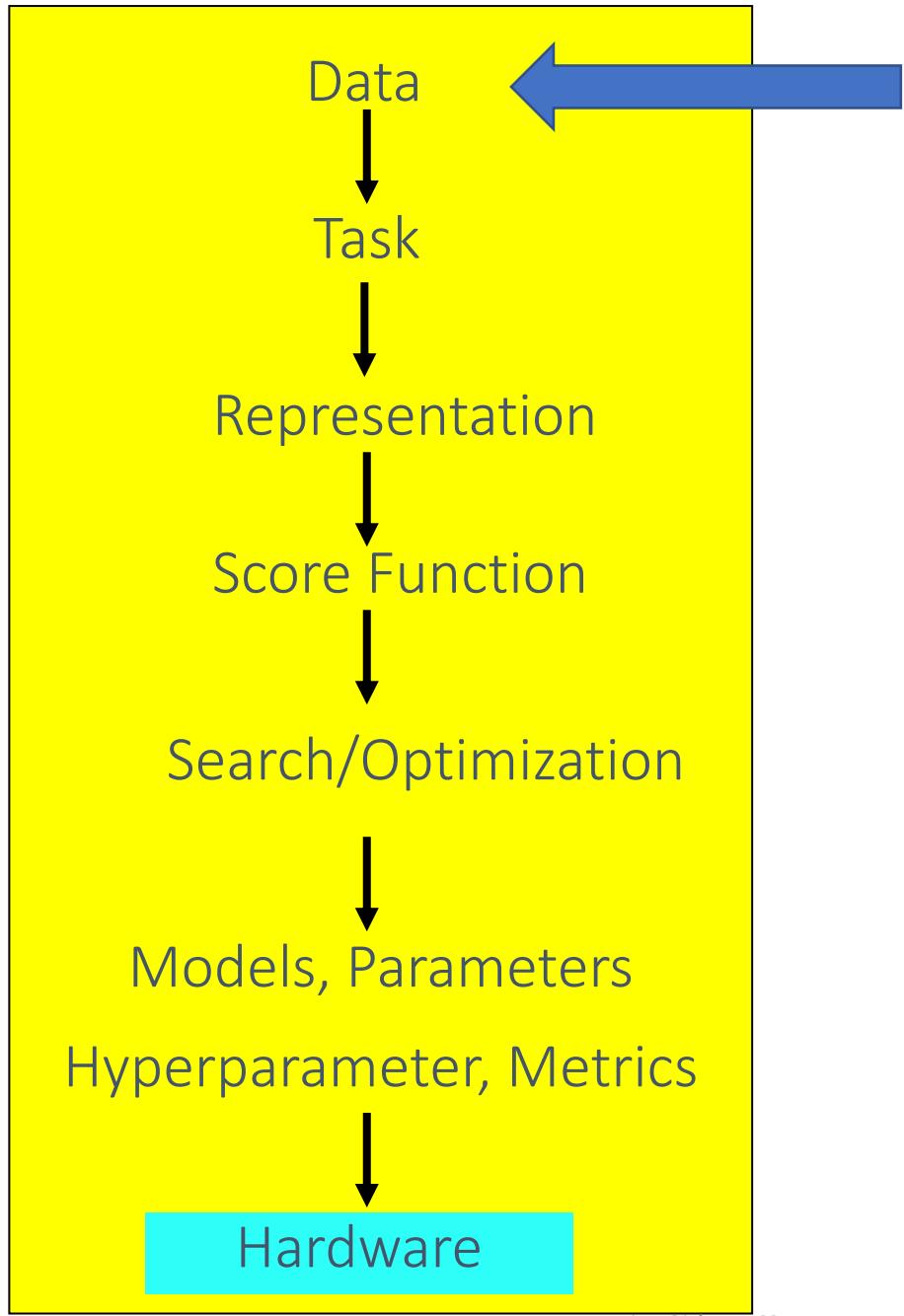
- 1. CNN / Residual / Memory
- 2. RNN / Attention / Seq2Seq / Transformer ...

- 1. Autoencoder / self supervised training
- 2. Generative Adversarial Networks (GAN)
- 3. Learning to optimize /Learning to search architecture (AutoML)

- 1. Validate / Evade / Test / Verify
- 2. Understand DNNs

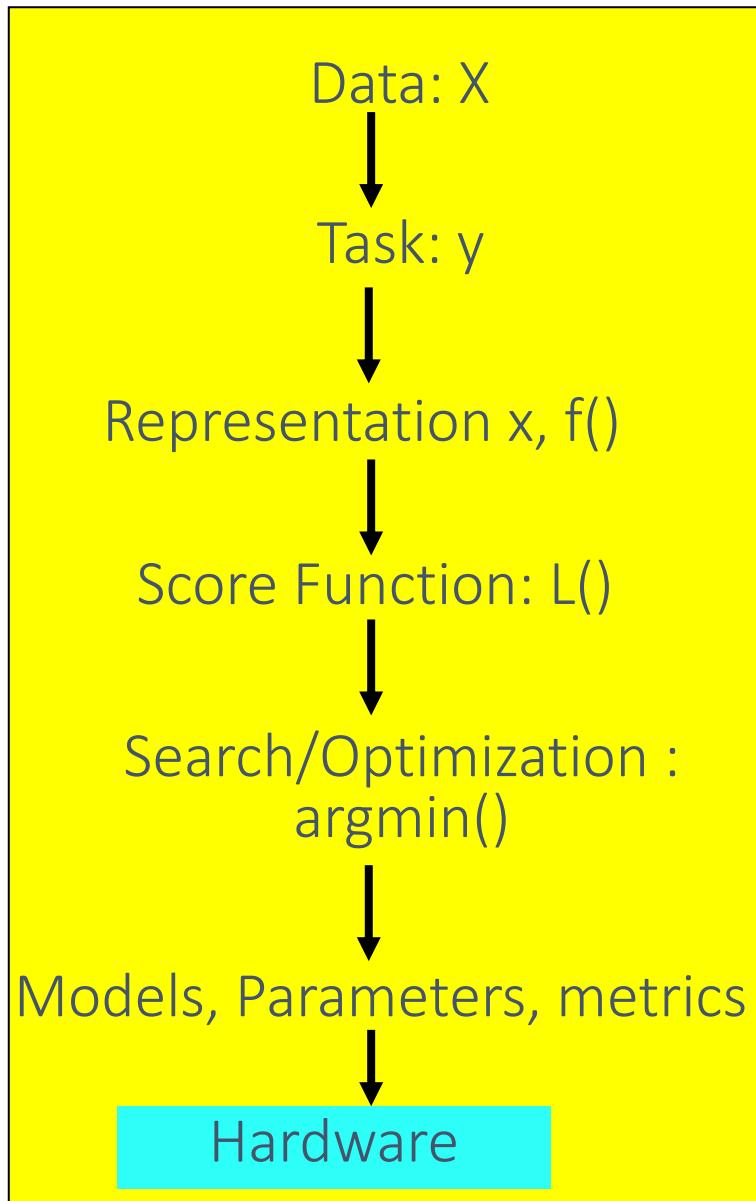
- 1. Model Compression / Efficient Net

# Deep Learning in a Nutshell



# Selected Trends

<https://qdata.github.io/deep2Read/>



- 1. DNN on graphs / trees / sets
- 2. NTM 4program induction

Recent Trend (1):  
Variants of Input, e.g.,  
Graphs, Trees, Sets

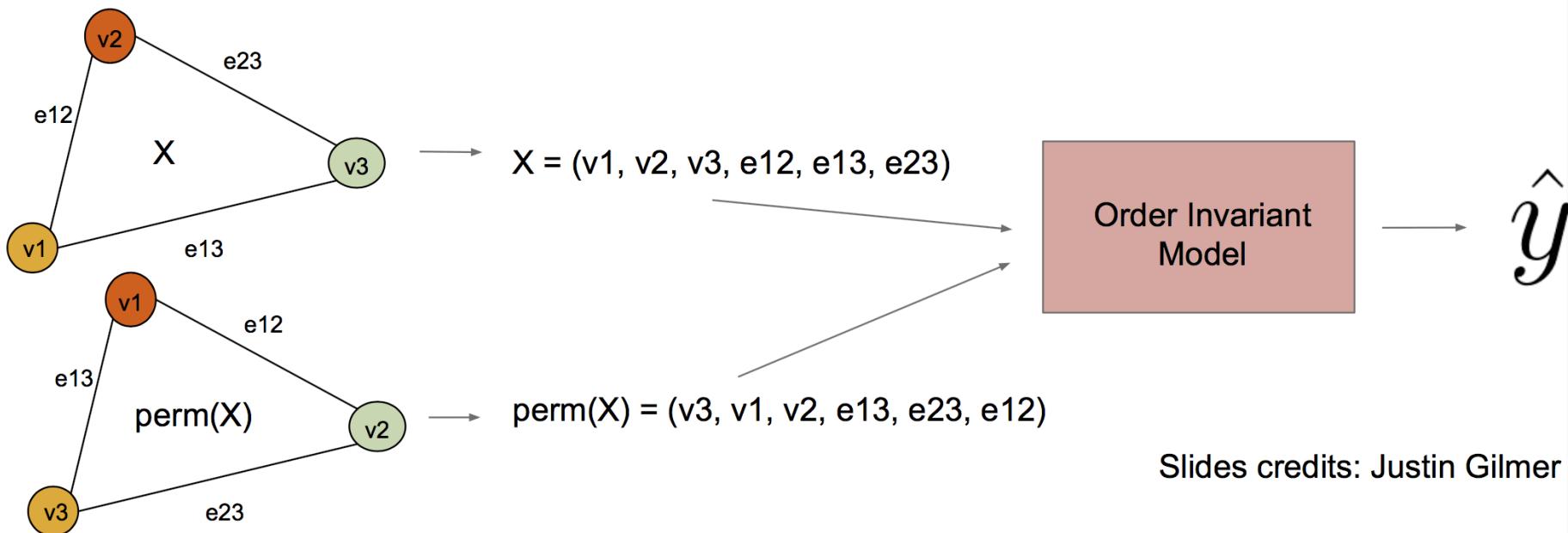


**Inputs and Outputs**

# Inductive Bias for Graphs

Arch

- If we have a graph on N nodes, there are  $N!$  possible orderings of the nodes.
- Ideally want a model invariant to the order of nodes.



Slides credits: Justin Gilmer

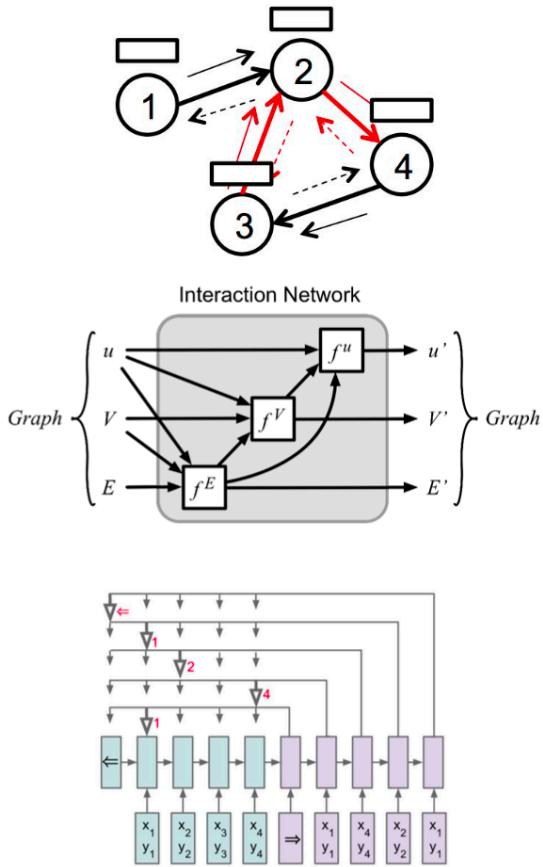
# Geometric Deep Learning on Graphs and Manifolds, NIPS 2017 Tutorial

Graph Nets (GNs) are a class of models that:

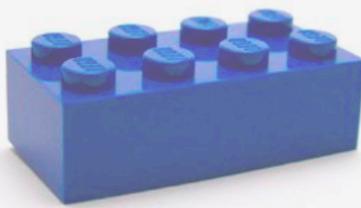
- Use graphs as inputs and/or outputs and/or latent representation
- Manipulate graph-structured representations
- Reflect relational structure
- Share model components across entities and relations

Examples include:

- Graph Neural Networks (*Scarselli et al 07; 08*)
- Recursive Neural Networks (*Goller et al 96*)
- Pointer Networks (*Vinyals et al 2015*)
- Graph Convolutional Networks (*Bruna et al 2013; Duvenaud et al 15; Henaff et al 15; Kipf & Welling 16; Defferrard et al 17*)
- Gated Graph Neural Networks (*Li et al 15*)
- Interaction Networks (*Battaglia et al 2016; Raposo et al 2017;*)
- Message Passing Networks (*Gilmer et al. 2017*)



# Recent Trend (2): Reasoning Tasks in the form of Symbolic input/ outputs / e.g., Program Induction



## Inputs and Outputs:

- Discrete symbols, (e.g. the program itself)
- Program execution traces
- Program I/O pairs

These can also be mixed with perceptual data.

## Architectures:

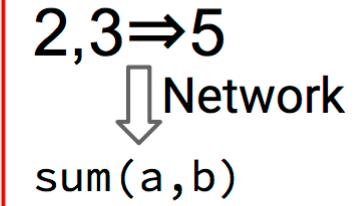
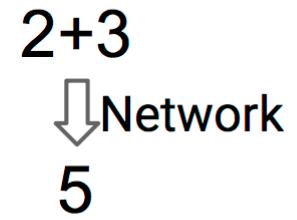
- (Mostly) recurrent
- Sometimes including ConvNets as a visual front-end.

## Losses:

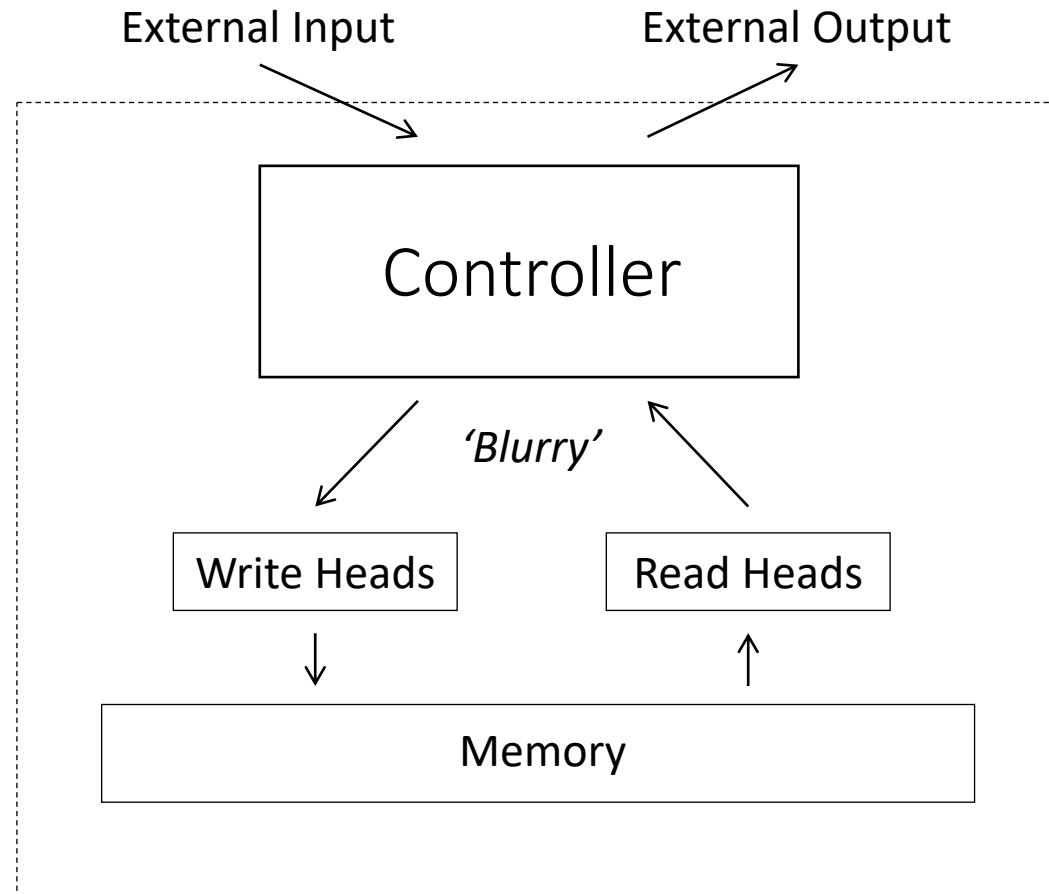
- Differentiable, predicting discrete program outputs or code itself: softmax cross entropy.
- Not differentiable: RL

# Neural Program Induction - Research Landscape

- Neural network is the program:
  - [Learning to Execute](#), [Neural Turing Machine](#), [Neural GPU](#), [Neural RAM](#), [Neural Programmer-Interpreter](#), [Neural Task Programmer](#), [Differentiable Forth Interpreter](#)
- Neural network generates source code :
  - [DeepCoder](#), [RobustFill](#), [Neural Inductive Logic Programming](#)
- Probabilistic programming with neural networks:
  - [TerpreT](#), [Edward](#), [Picture](#)



# Neural Turing Machines

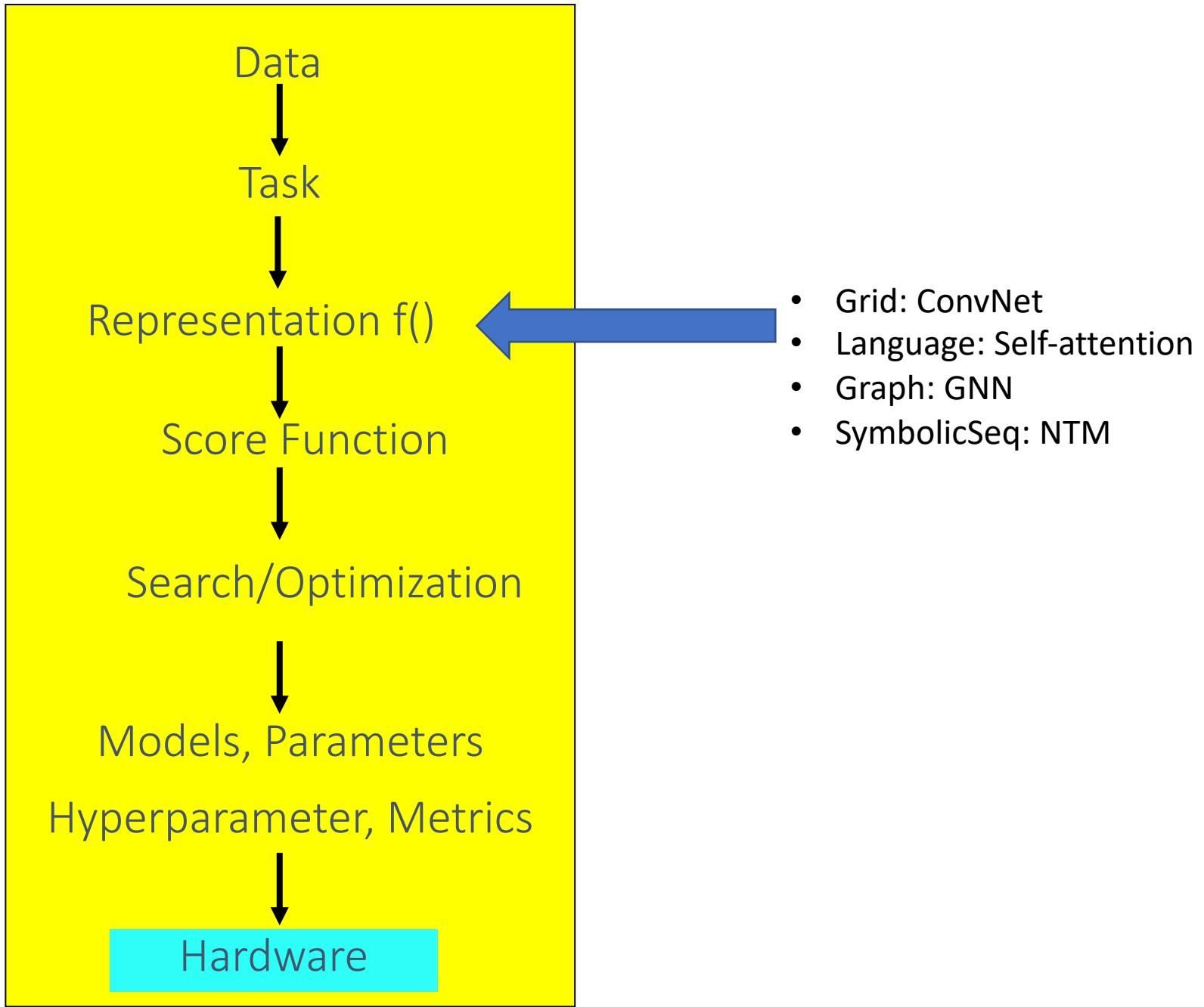


Neural Turing Machines, Graves et. al., arXiv:1410.5401

# Task with Sequential Symbolic Form

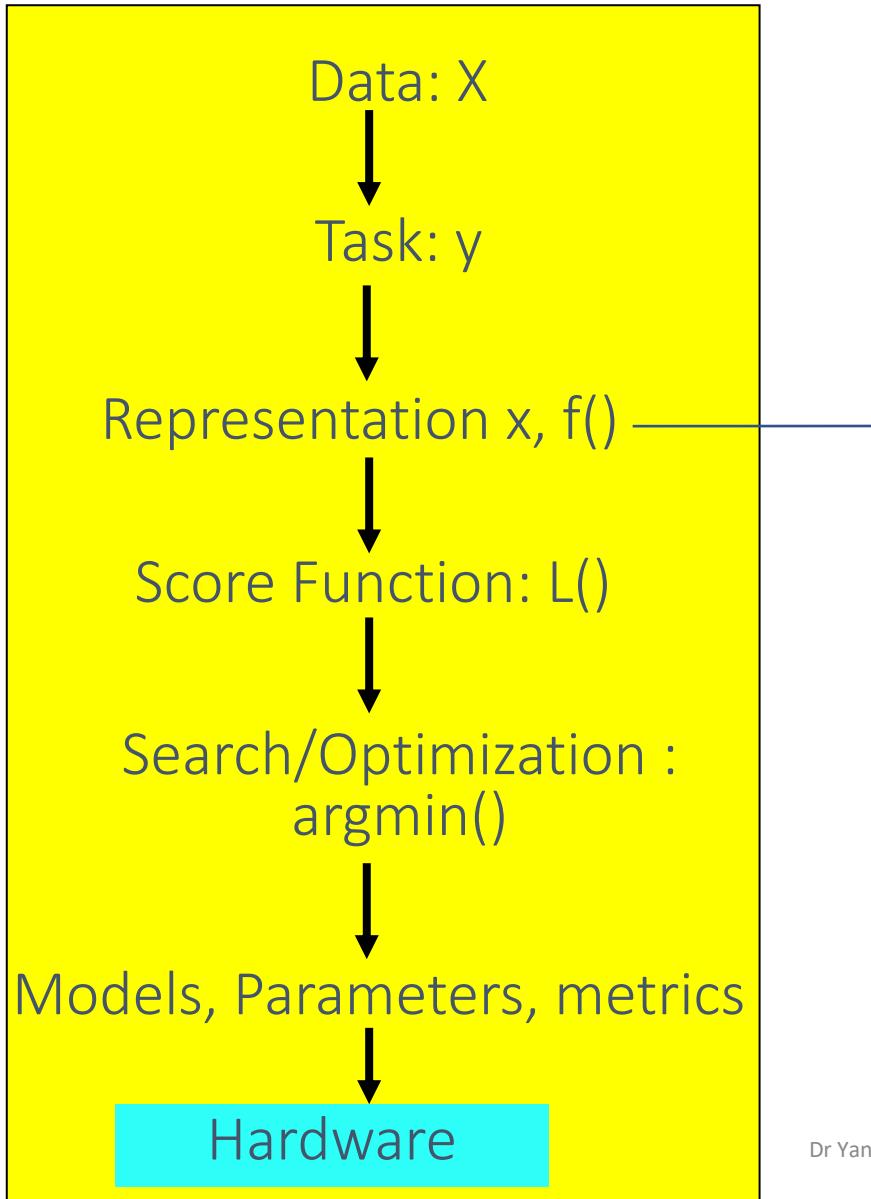
- Computer Programs , ...
- Sequence decision making, e.g., games with symbolic

# Deep Learning in a Nutshell



# Selected Trends

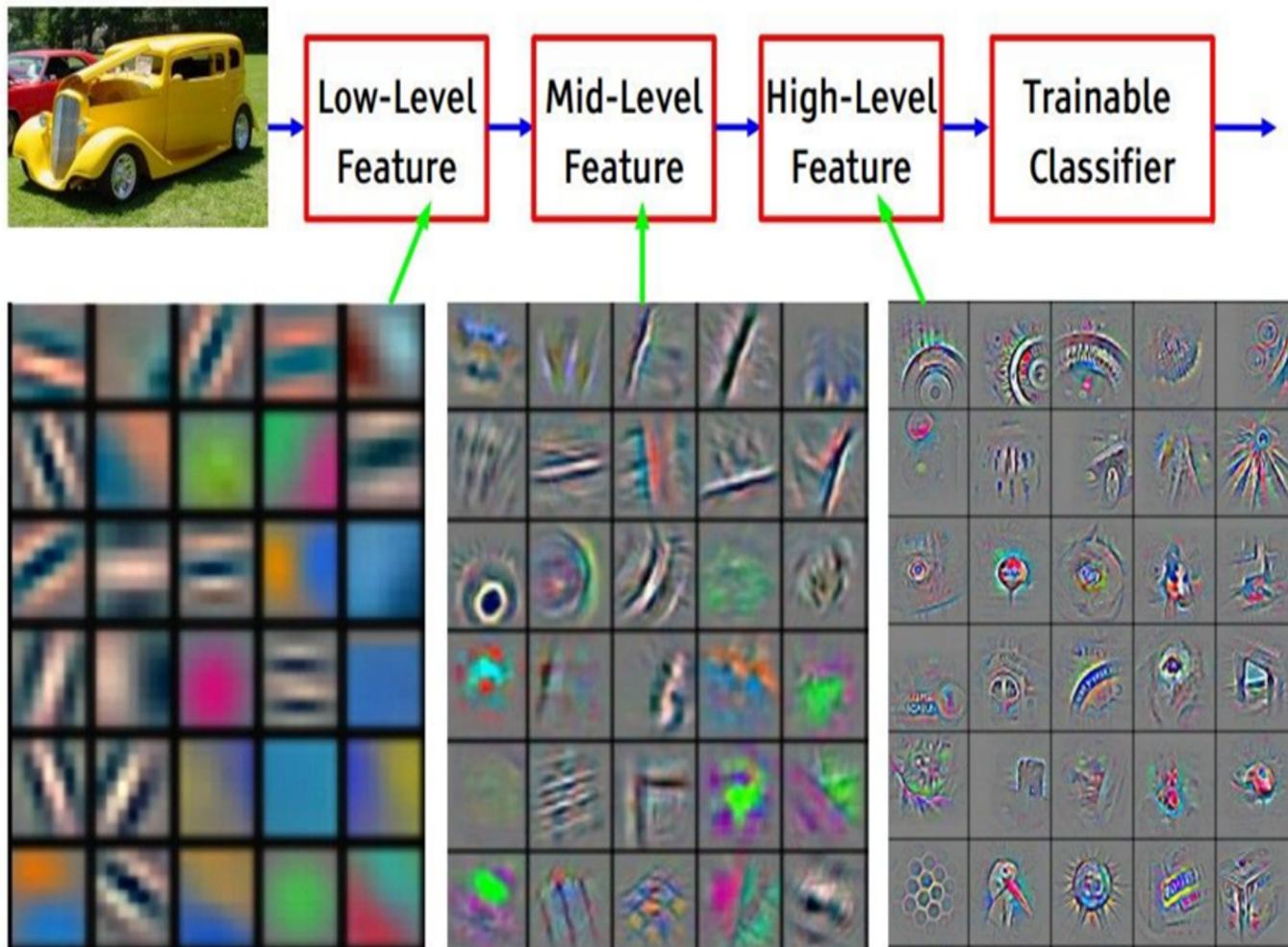
<https://qdata.github.io/deep2Read/>



- 1. CNN / Residual / Memory
- 2. RNN / Attention / Seq2Seq / Transformer ...

# Convolutional Neural Networks

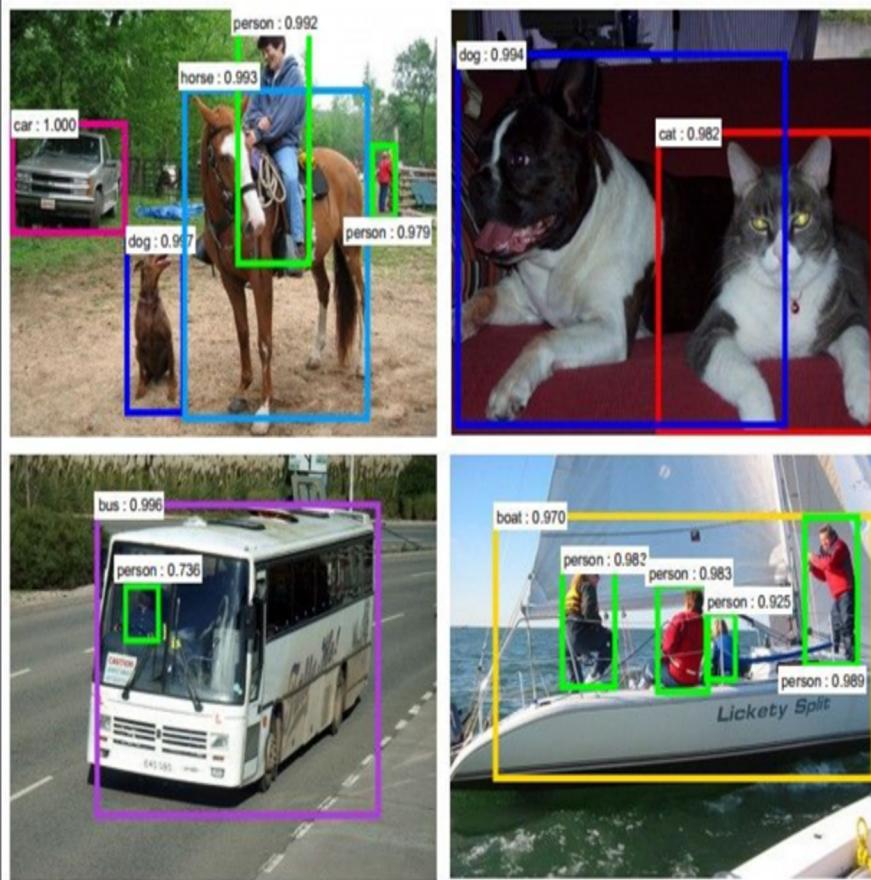
[From recent Yann LeCun slides]



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

# ConvNets are everywhere

## Detection



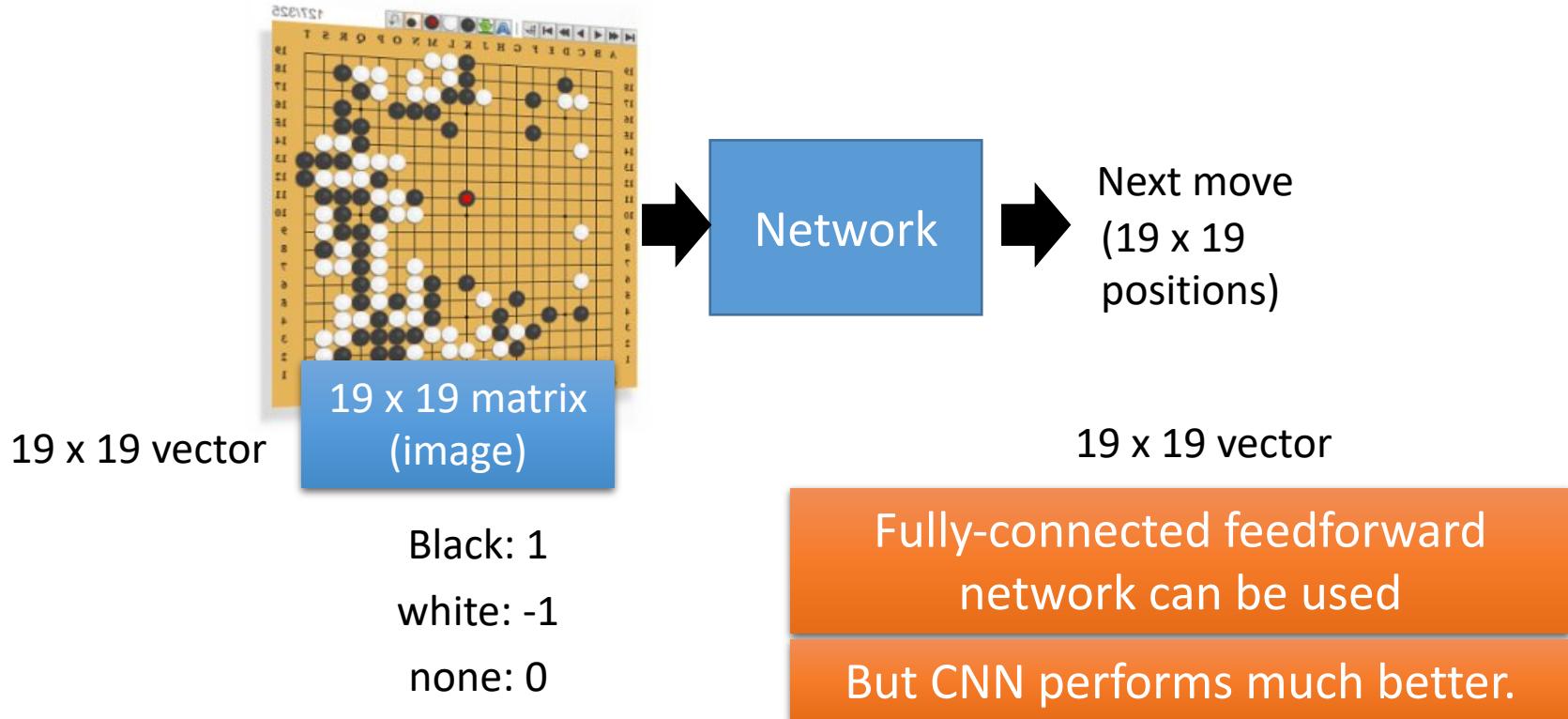
[Faster R-CNN: Ren, He, Girshick, Sun 2015]

## Segmentation

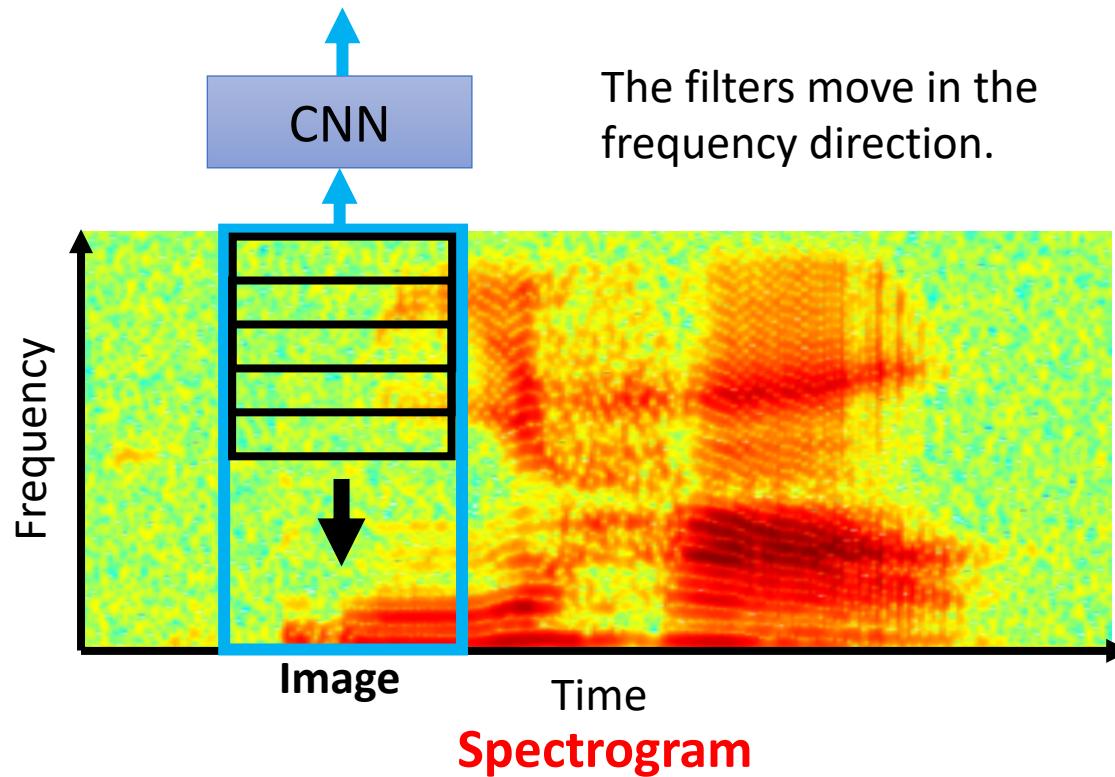


[Farabet et al., 2012]

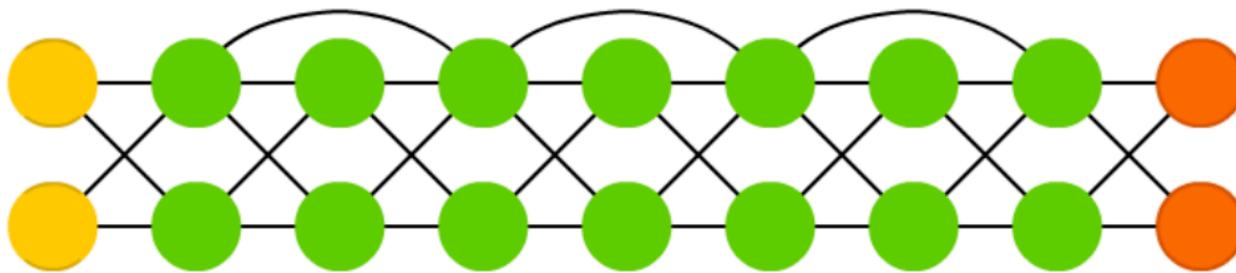
# More Application: Playing Go



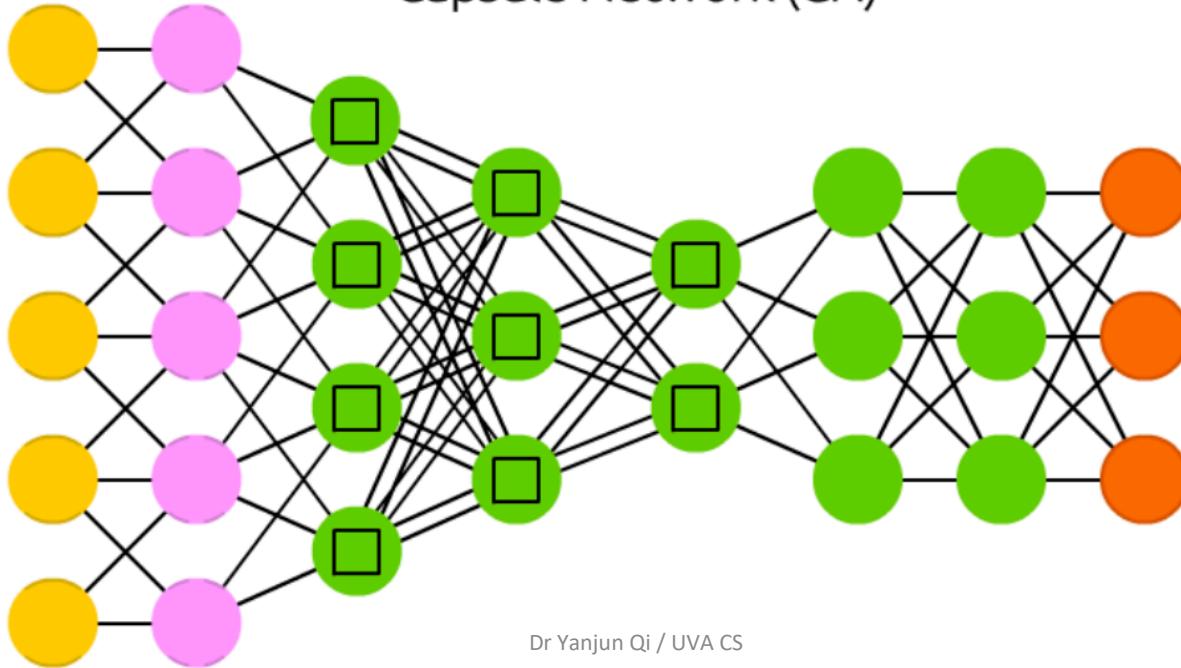
# More Application: Speech



## Deep Residual Network (DRN)



## Capsule Network (CN)

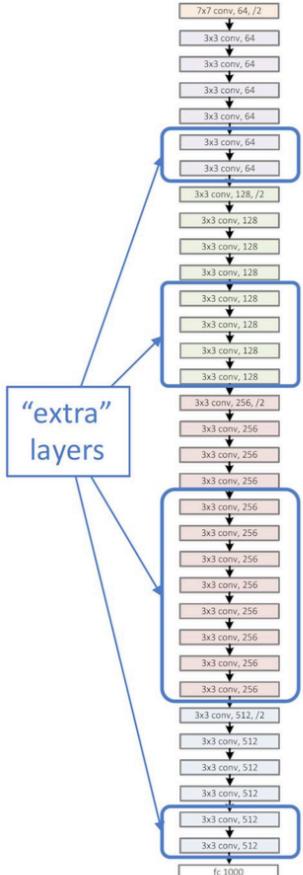
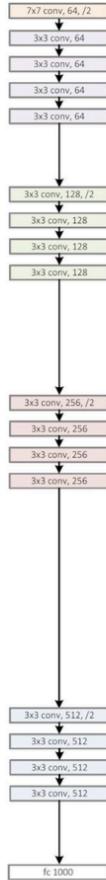


## Residual Trick:

# Residual/Skip Connections

Arch

a shallower  
model  
(18 layers)



a deeper  
counterpart  
(34 layers)

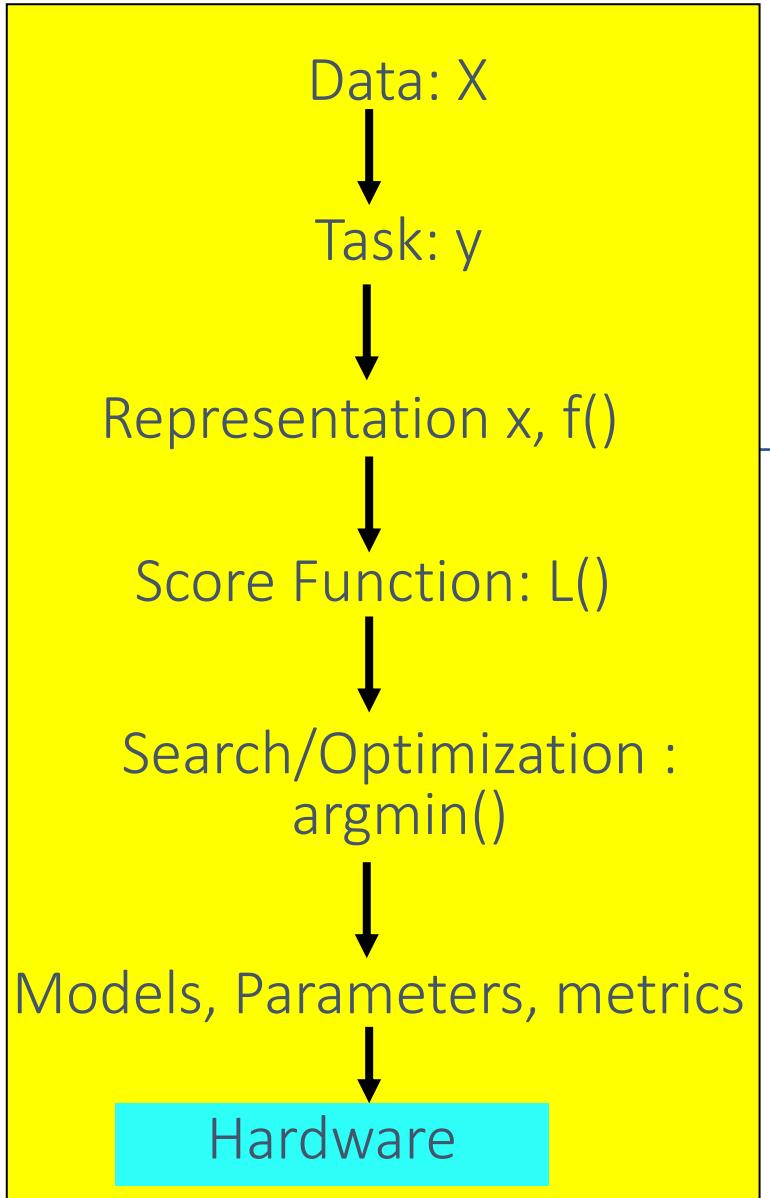
- Richer solution space
- A deeper model should not have **higher training error**
- A solution *by construction*:
  - original layers: copied from a learned shallower model
  - extra layers: set as **identity**
  - at least the same training error
- **Optimization difficulties**: solvers cannot find the solution when going deeper...

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.



# Selected Trends

<https://qdata.github.io/deep2Read/>



- 1. CNN / Residual / Memory
- 2. RNN / Attention / Seq2Seq / Transformer ...



THE  
TRANSFORMER



OpenAI  
Transformer

ELMo: Embeddings from Language Models  
Pre-trained biLSTM for contextual embedding

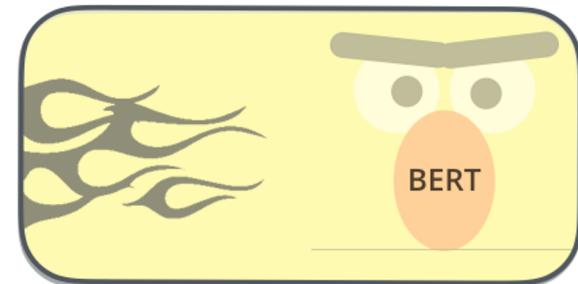
BERT: Bidirectional Encoder Representations  
from Transformers  
Pre-trained transformer encoder for  
sentence embedding

10/27/20

Based: Dr. Yangqiu Song's slides

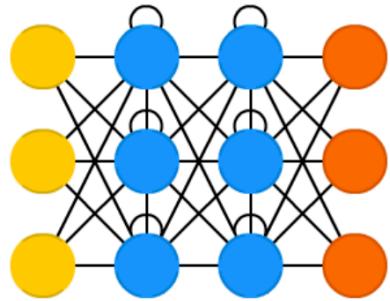


# Notable pre-trained NLP models

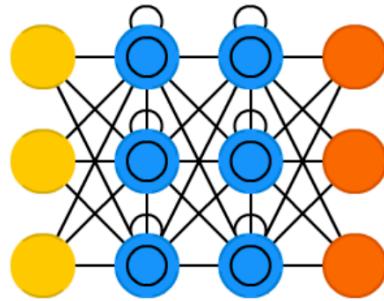


- BERT ([Google](#))
- XLNet ([Google/CMU](#))
- RoBERTa ([Facebook](#))
- DistilBERT ([HuggingFace](#))
- CTRL ([Salesforce](#))
- GPT-2 ([OpenAI](#))
- ALBERT ([Google](#))
- Megatron ([NVIDIA](#))

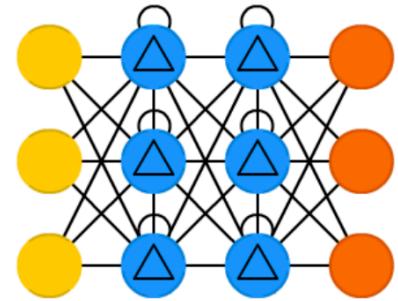
Recurrent Neural Network (RNN)



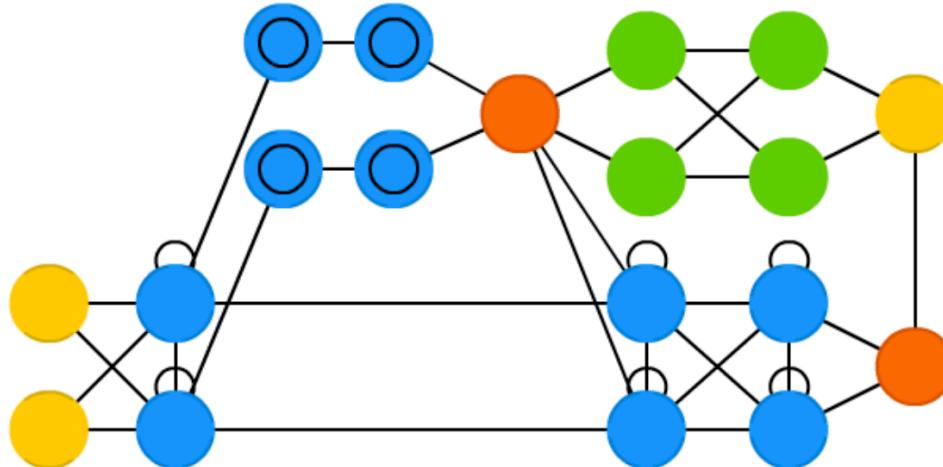
Long / Short Term Memory (LSTM)

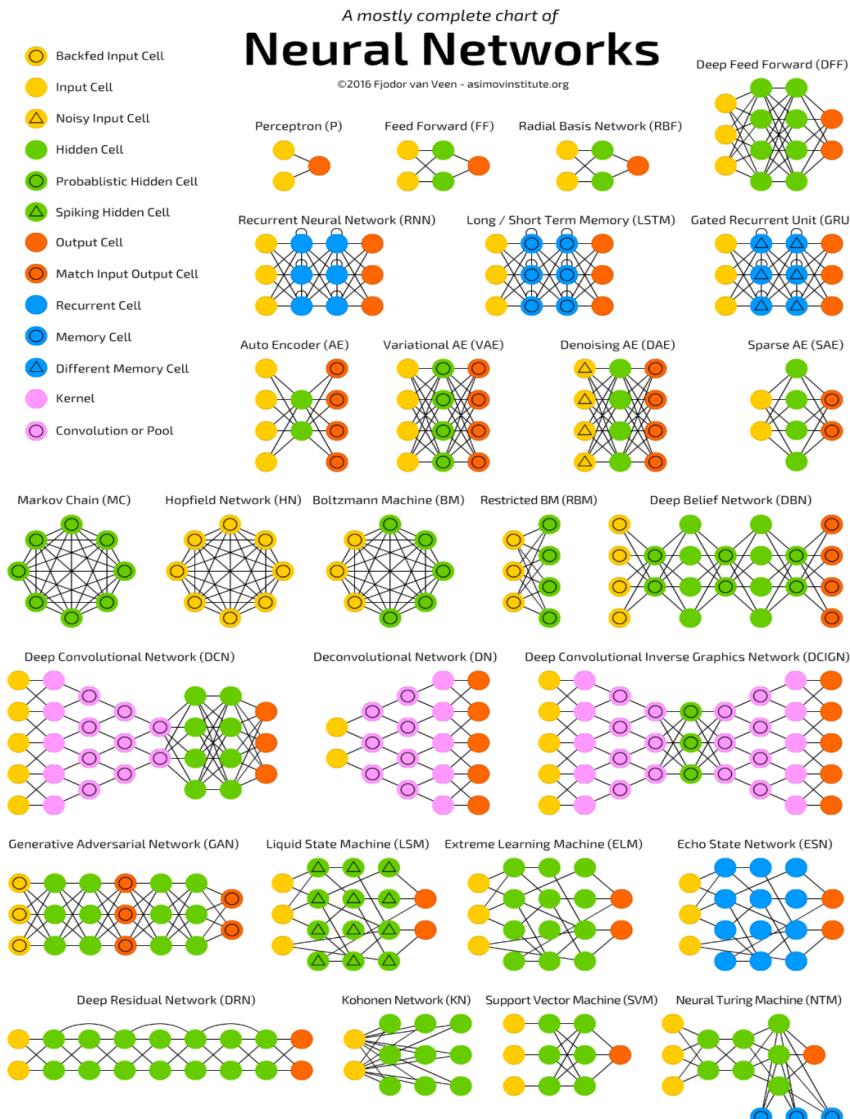


Gated Recurrent Unit (GRU)



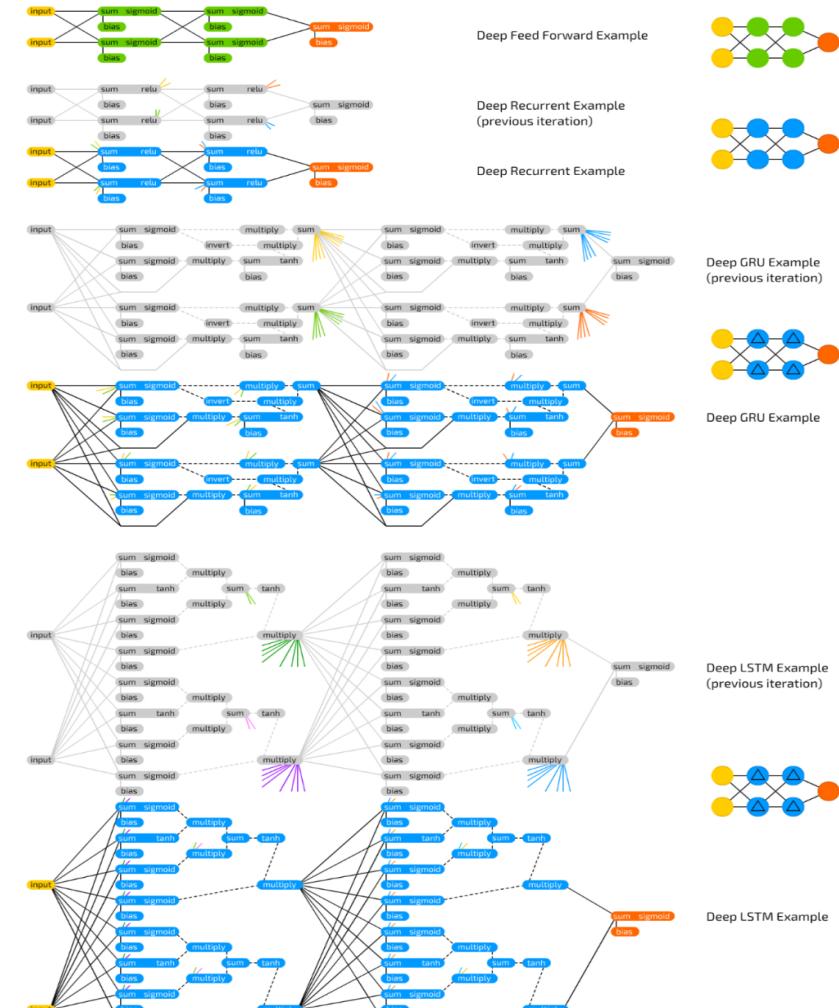
Attention Network (AN)





An informative chart to build  
**Neural Network Graphs**

©2016 Fjodor van Veen - asimovinstitute.org



# UVA CS 4774: Machine Learning



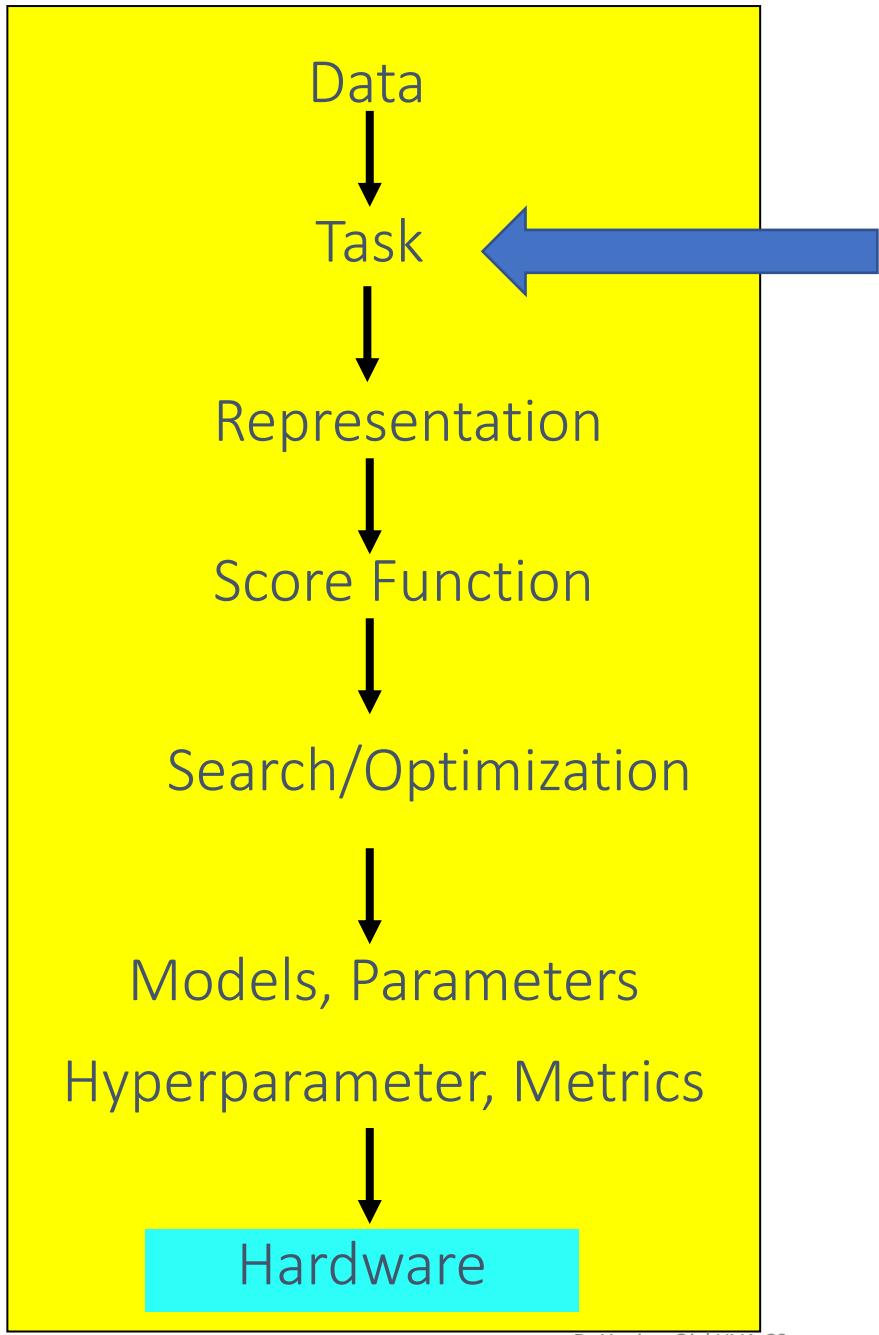
## S3: Lecture 19: Recent Deep Neural Networks: A Quick Overview

Dr. Yanjun Qi

University of Virginia  
Department of Computer Science

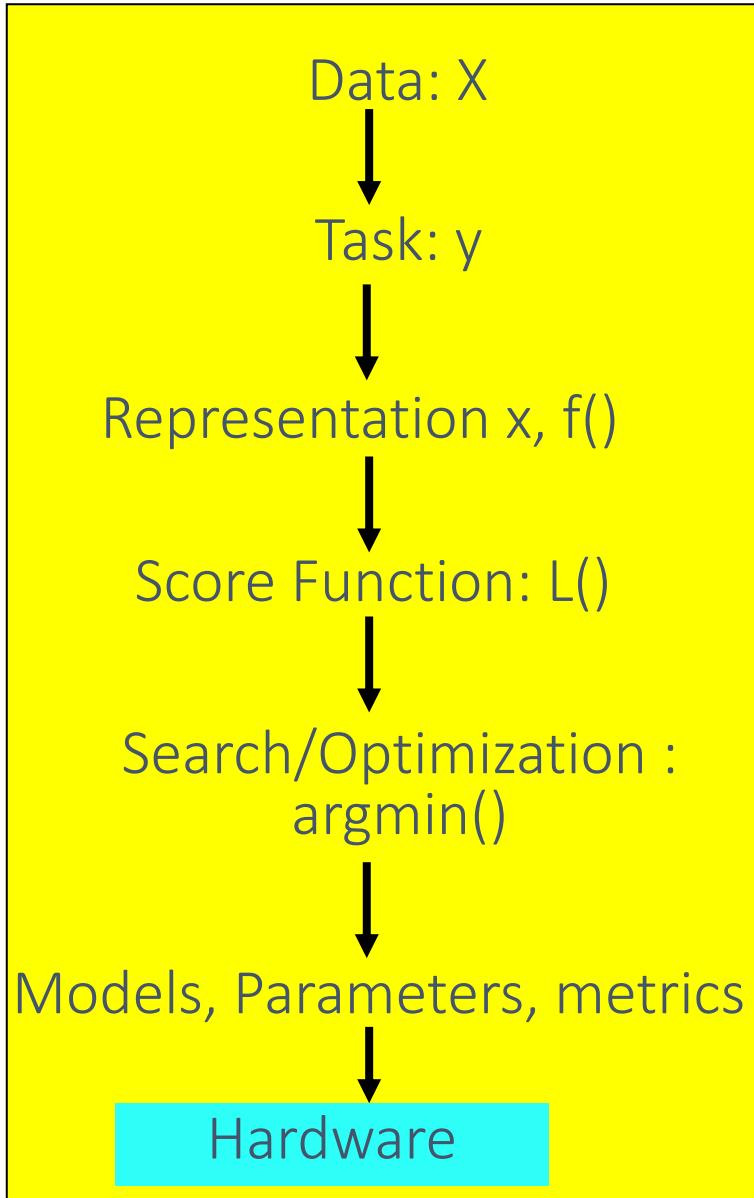
Module II

# Deep Learning in a Nutshell



# Selected Trends

<https://qdata.github.io/deep2Read/>



- 1. Deep Generative models/ DeepFake
- 2. Deep reinforcement learning
- 3. Few-shots / Meta learning / AGI?

# Recent Trend (3): Deep Generative Models

## Generative models - Research Landscape

- Latent variable models ([VAE](#), [DRAW](#))
- Implicit ([GAN](#), [GMMN](#), [Progressive GAN](#))
- Transform ([NICE](#), [IAF](#), [Real NVP](#))
- **Autoregressive** ([NADE](#), [MADE](#), [RIDE](#), [PixelCNN](#), [WaveNet](#))

UAI 2017 [Tutorial](#) on Deep Generative Models.

NIPS 2016 [Tutorial](#) on Generative Adversarial Networks

# Why Generative Models?

- Excellent test of ability to use high-dimensional, complicated probability distributions
- Simulate possible futures for planning or simulated RL
- Missing data
  - Semi-supervised learning
- Multi-modal outputs
- Realistic generation tasks

# Image Super-Resolution

bicubic  
(21.59dB/0.6423)



SRResNet  
(23.53dB/0.7832)



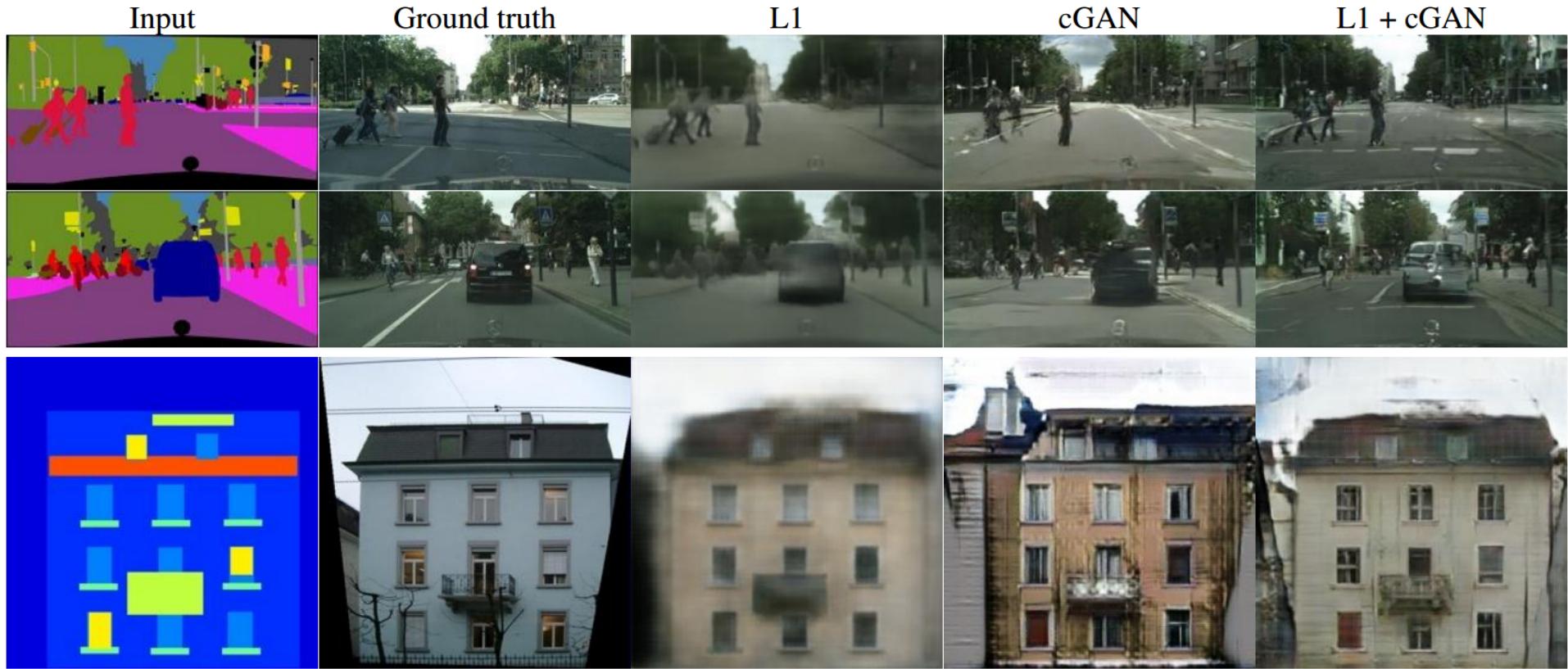
SRGAN  
(21.15dB/0.6868)



original



# Label2Image



# Edges2Image



# Text2Image

this small bird has a pink breast and crown, and black primaries and secondaries.



this magnificent fellow is almost all black with a red crest, and white cheek patch.



the flower has petals that are bright pinkish purple with white stigma



this white and yellow flower have thin white petals and a round yellow stamen



# DeepFake (Generation and Detection)

- A deepfake is generally understood to be a video in which the face of one person has been swapped with the face of another person
- Many variations on this (face swap, puppet-master, lip-sync)

DEEP FAKE

Coarse styles ( $4^2 - 8^2$ )

Middle styles ( $16^2 - 32^2$ )

Fine styles ( $64^2 - 1024^2$ )

IT News

FACEBOOK USES DEEFAKE-LIKE TECH TO ANONYMISE FACES IN VIDEOS

Dr Yanjun Qi / UVA CS

<https://github.com/iperov/DeepFaceLab>



De-Aged the face



# Deepfake history

**June 2016**

Face2Face paper released

**July 2017**

Synthesizing Obama paper released (audio lip syncing)

**Winter 2017**

r/deepfakes subreddit created

**Feb 2018**

r/deepfakes subreddit banned

**April 2018**

Jordan Peele Obama PSA deepfake released

**June 2018**

In Ictu Oculi: Exposing AI Generated Fake Face Videos by Detecting Eye Blinking paper released

**Sept 2018**

MesoNet: a Compact Facial Video Forgery Detection Network paper released

**April 2019**

FaceForensics++ paper and dataset released

**May 2019**

Few Shot Adversarial Learning of Realistic Neural Talking Heads Model paper released

**June 2019**

Text-Based Editing of Talking-head Video paper released

**June 2019**

Mark Zuckerberg deepfake released

# Recent Trend (4): Deep Reinforcement Learning

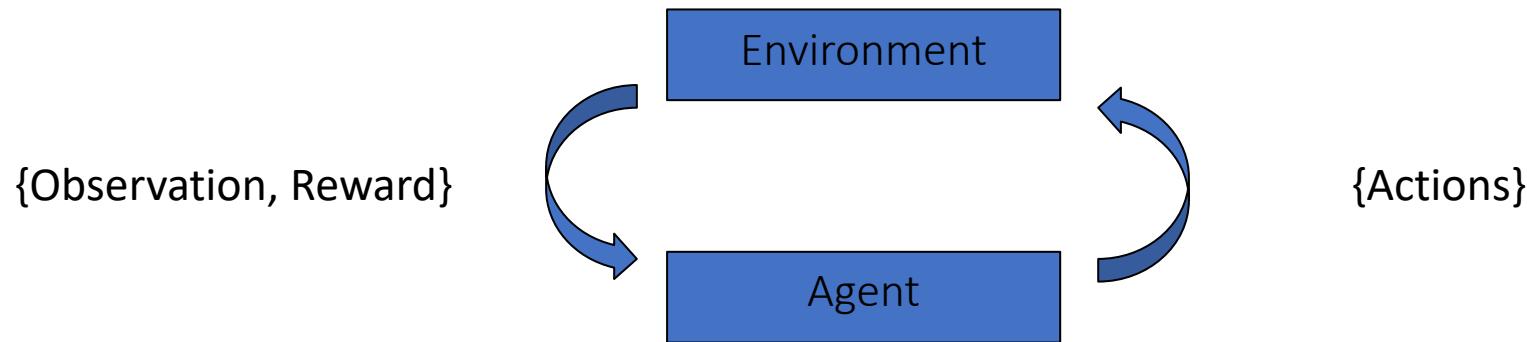
## 10 Breakthrough Technologies 2017

**T**hese technologies all have staying power. They will affect the economy and our politics, improve medicine, or influence our culture. Some are unfolding now; others will take a decade or more to develop. But you should know about all of them right now.

**MIT  
Technology  
Review**

# Reinforcement Learning (RL)

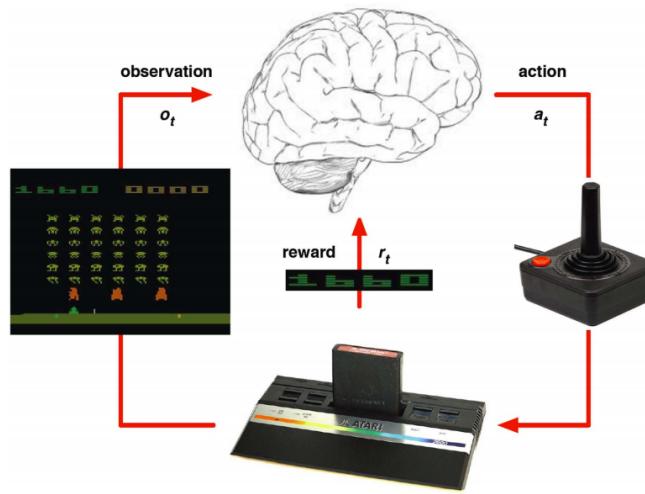
- What's Reinforcement Learning?



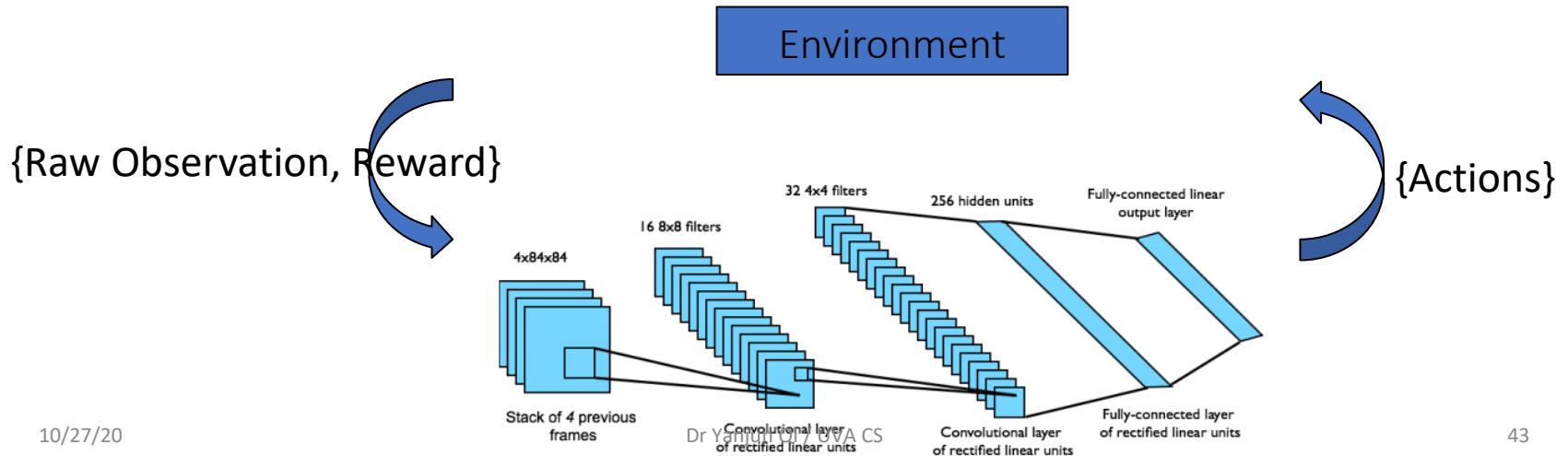
- Agent interacts with an environment and learns by maximizing a scalar reward signal
- No labels or any other supervision signal.
- Previously suffering from hand-craft states or representation.

# Deep Reinforcement Learning

- Human

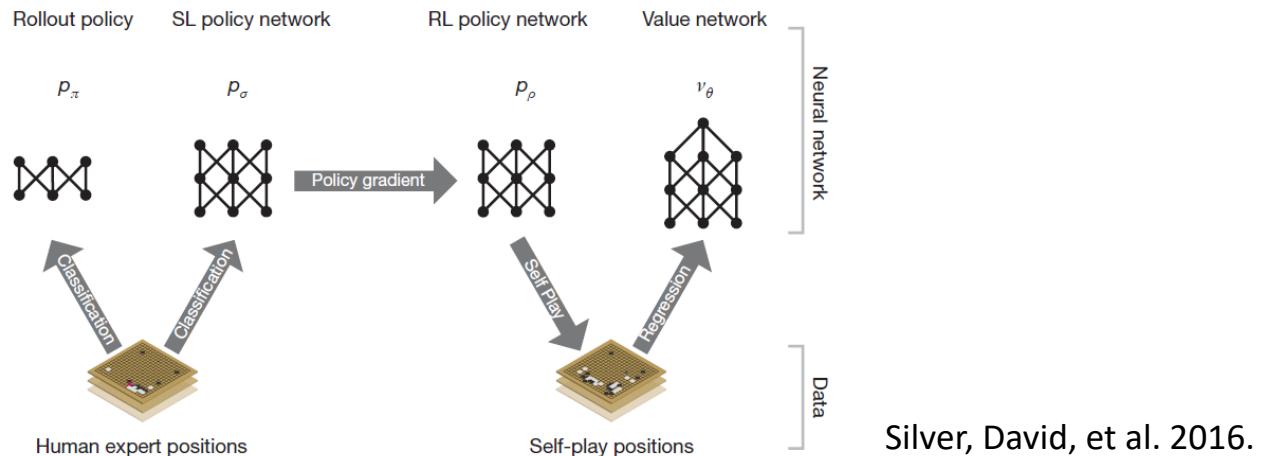


- So what's **DEEP RL**?



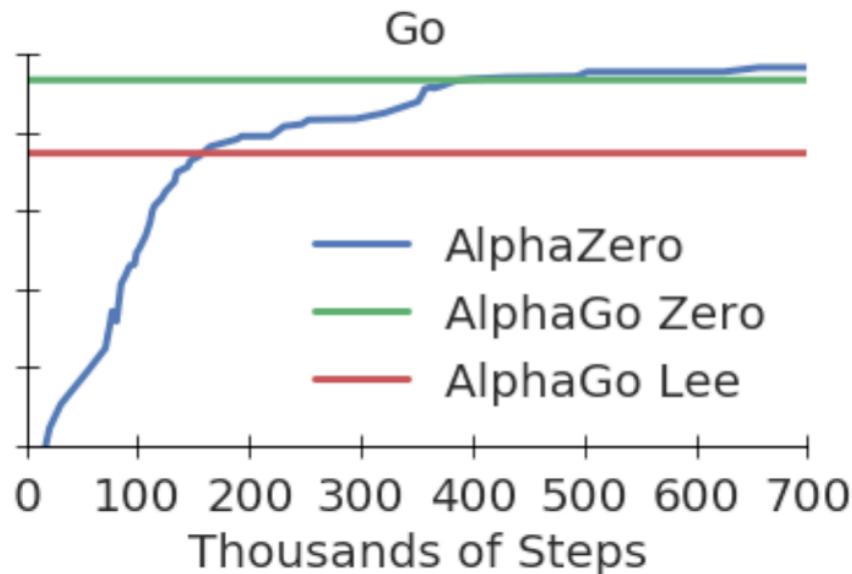
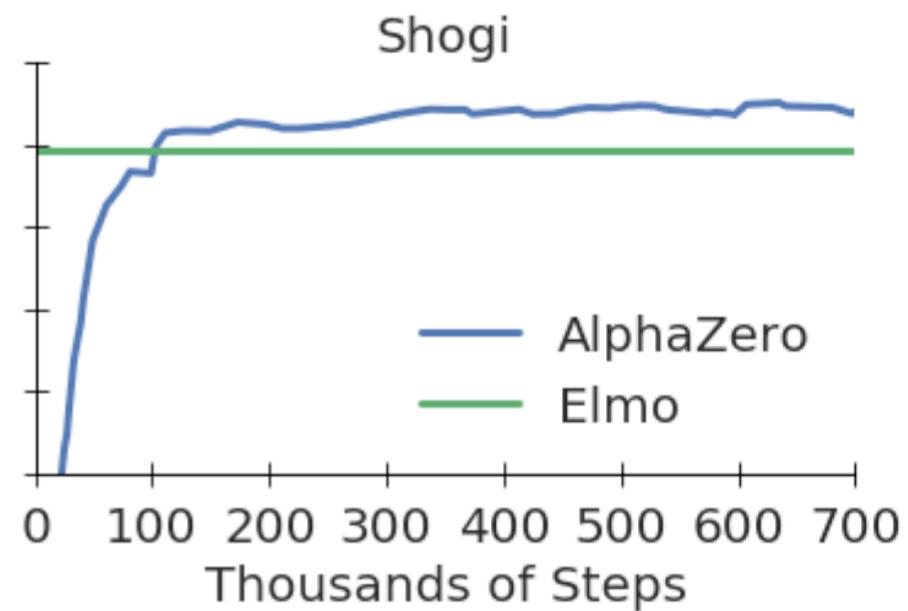
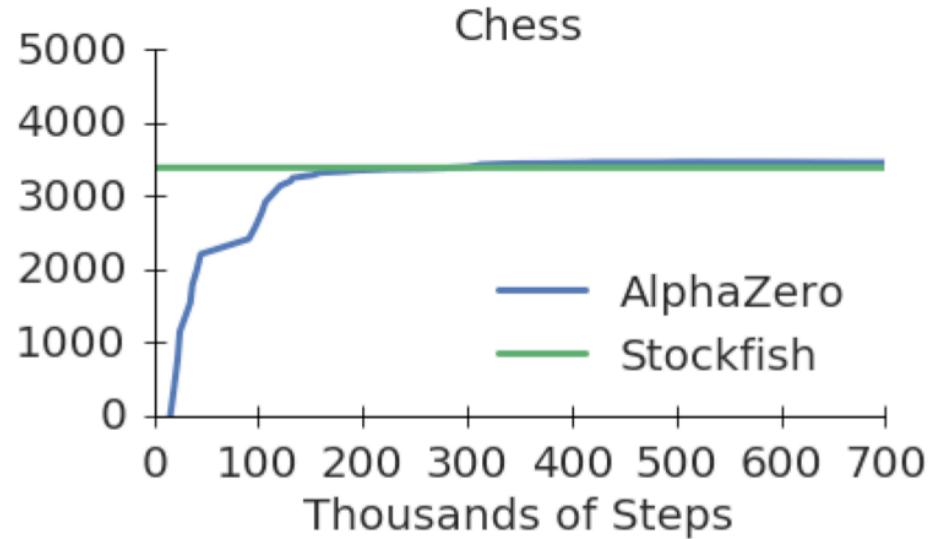
# AlphaGO: Learning Pipeline

- Combine Supervised Learning (SL) and RL to learn the search direction in Monte Carlo Tree Search



Silver, David, et al. 2016.

- SL policy Network
  - Prior search probability or potential
- Rollout:
  - combine with MCTS for quick simulation on leaf node
- Value Network:
  - Build the Global feeling on the leaf node situation





latest

Search docs

## USER DOCUMENTATION

Introduction

Installation

Algorithms

Running Experiments

Experiment Outputs

Plotting Results

## INTRODUCTION TO RL

Part 1: Key Concepts in RL

Part 2: Kinds of RL Algorithms

Part 3: Intro to Policy Optimization

## RESOURCES

Spinning Up as a Deep RL Researcher

Key Papers in Deep RL

10/27/20

# Benchmarks for Spinning Up Implementations

## Table of Contents

- [Benchmarks for Spinning Up Implementations](#)
  - [Performance in Each Environment](#)
    - [HalfCheetah: PyTorch Versions](#)
    - [HalfCheetah: Tensorflow Versions](#)
    - [Hopper: PyTorch Versions](#)
    - [Hopper: Tensorflow Versions](#)
    - [Walker2d: PyTorch Versions](#)
    - [Walker2d: Tensorflow Versions](#)
    - [Swimmer: PyTorch Versions](#)
    - [Swimmer: Tensorflow Versions](#)
    - [Ant: PyTorch Versions](#)
    - [Ant: Tensorflow Versions](#)
  - [Experiment Details](#)
  - [PyTorch vs Tensorflow](#)

We benchmarked the Spinning Up algorithm implementations in five environments from the [MuJoCo](#) Gym task suite: HalfCheetah, Hopper, Walker2d, Swimmer, and Ant.

## Performance in Each Environment

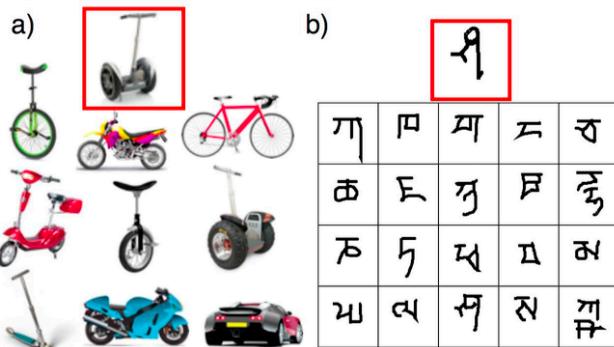
Dr Yanjun Qi / UVA CS

1 / 1

# Recent Trend (5): Meta Learning: Learning to Learn

# Learning to Learn

- What is Meta Learning / Learning to Learn?
  - Go beyond train/test from same distribution.
  - Task between train/test changes, so model has to “learn to learn”
- Datasets



Lake et al,  
2013, 2015



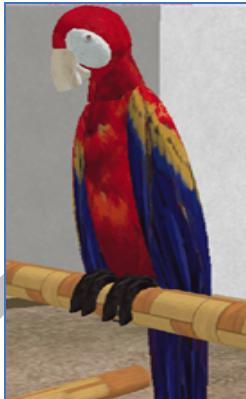
Chelsea Finn, UC Berkeley

How? learn to learn many other tasks

fig. from Duan et al. '17

# AGI (Artificial General Intelligence) versus Narrow AI

Intelligence...



## AGI:

- “The ability to achieve complex goals in complex environments using limited computational resources”
- Autonomy
- Practical understanding of self and others
- Understanding “what the problem is” as opposed to just solving problems posed explicitly by programmers
- Solving problems that were not known to the programmers

Narrow AI: game -playing, medical diagnosis, car-driving, etc.

# UVA CS 4774: Machine Learning



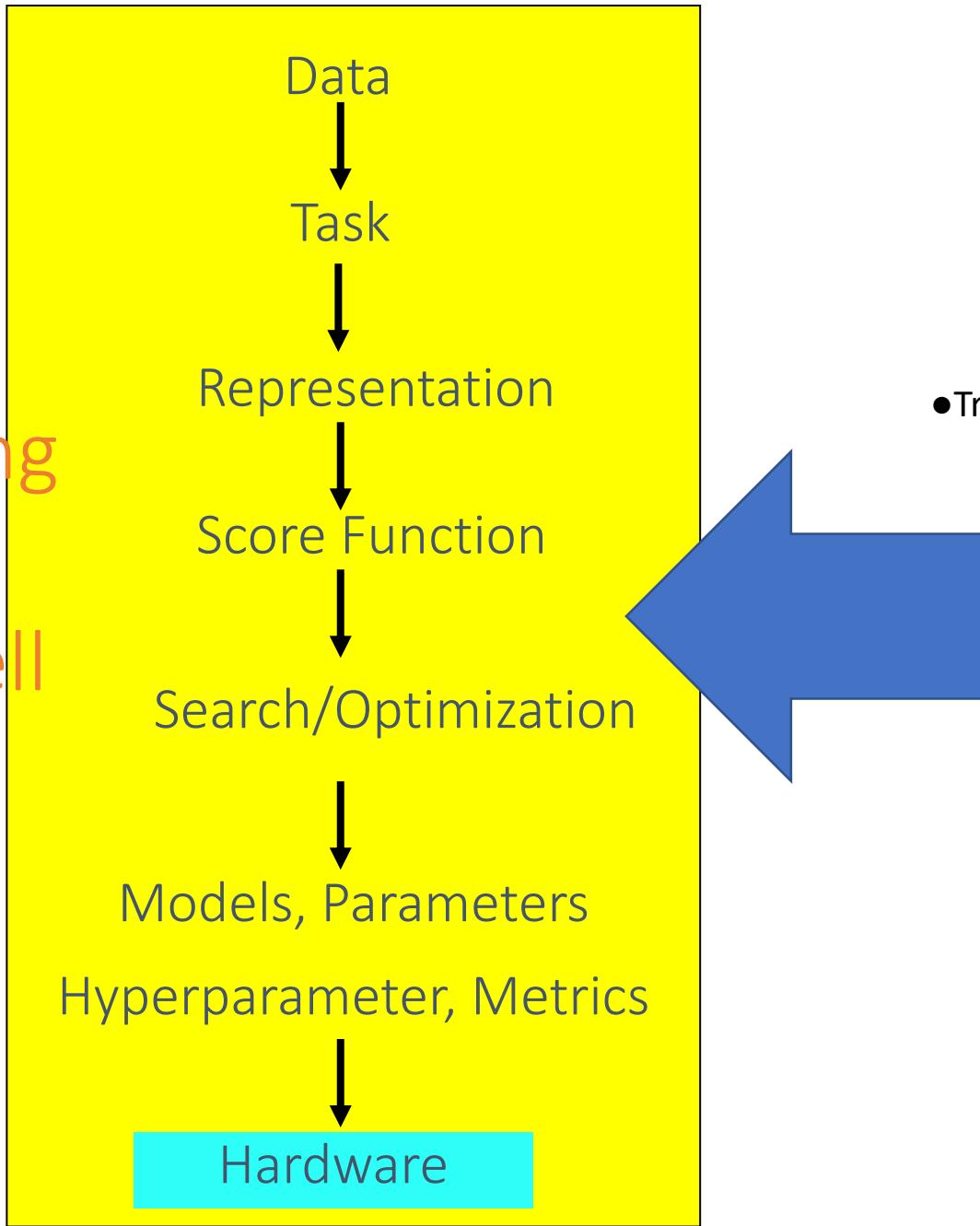
## S3: Lecture 19: Recent Deep Neural Networks: A Quick Overview

Dr. Yanjun Qi

University of Virginia  
Department of Computer Science

Module III

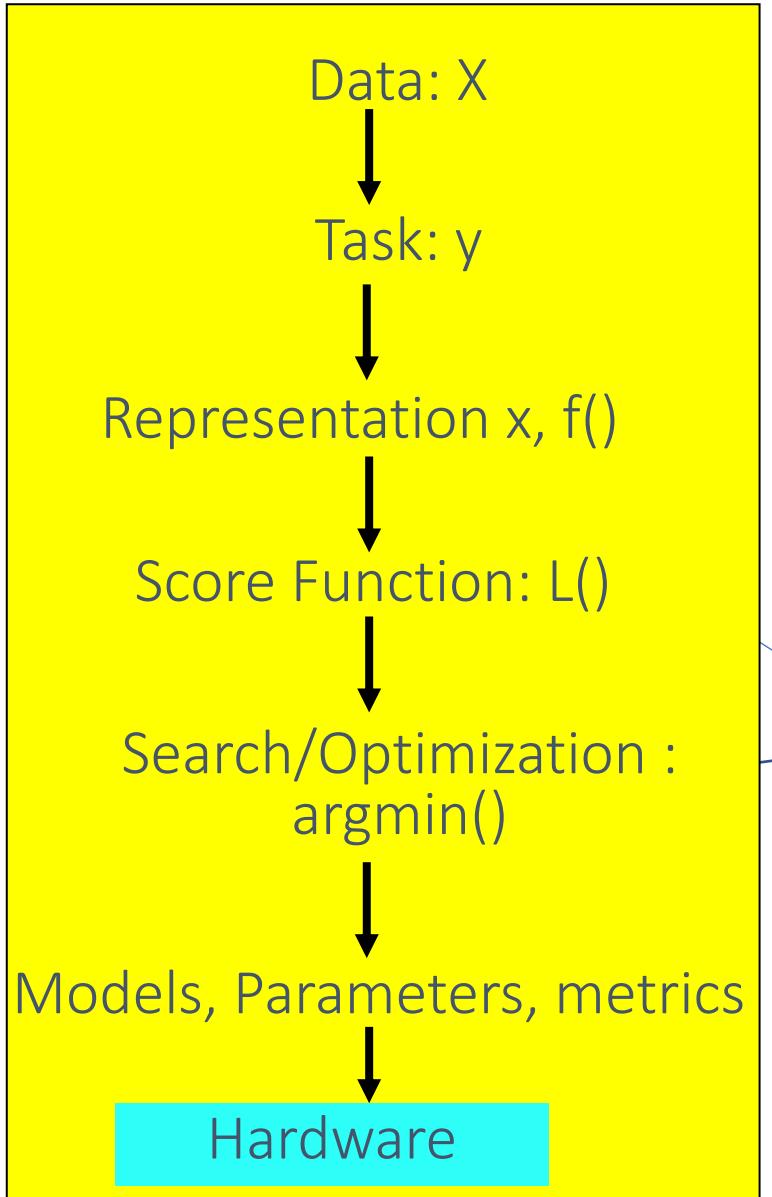
# Deep Learning in a Nutshell



- Training / Searching / Learning
  - With new losses
  - **With new optimization pipelines**
  - Scaling up with GPU, Scaling up with distributed optimization , e.g. Asynchronous SGD

# Selected Trends

<https://qdata.github.io/deep2Read/>



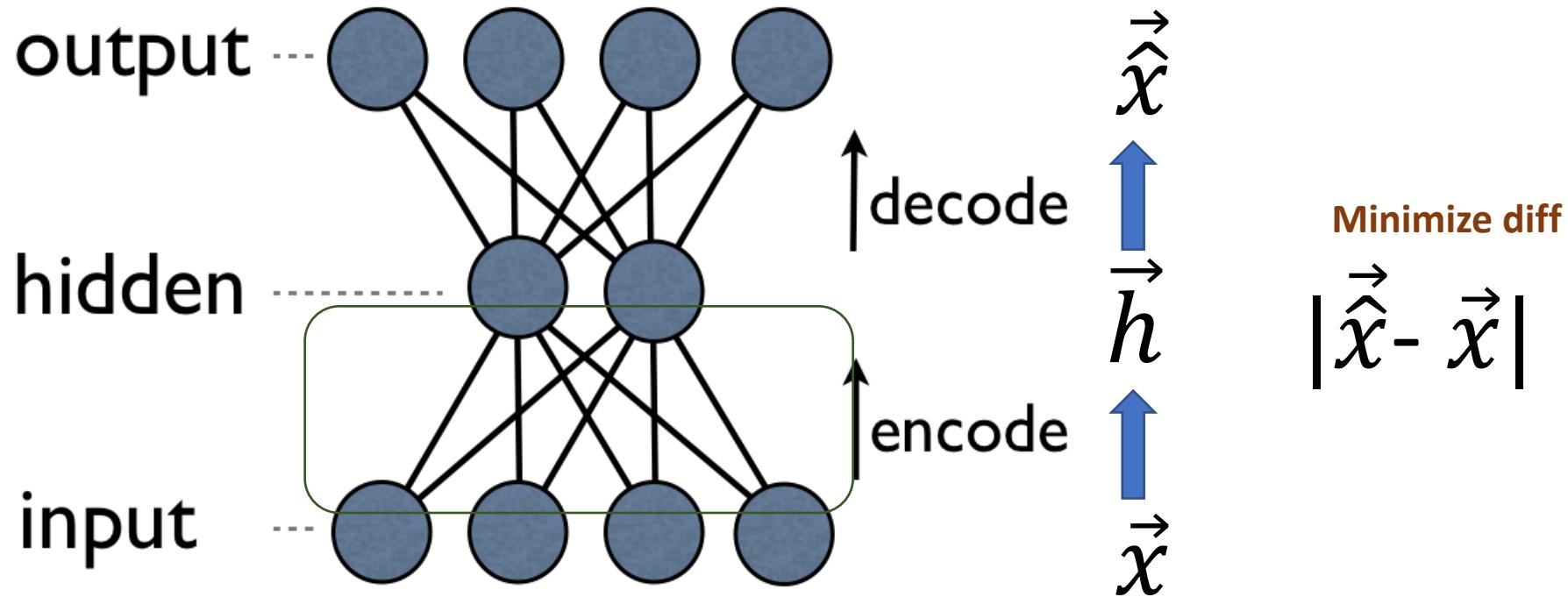
- 1. Autoencoder / self supervised training
- 2. Generative Adversarial Networks (GAN)
- 3. Learning to optimize / Learning to search architecture (AutoML)

# Recent Trend (6): Layer-wise pretraining / Auto-Encoder / Self-supervised



**Losses**

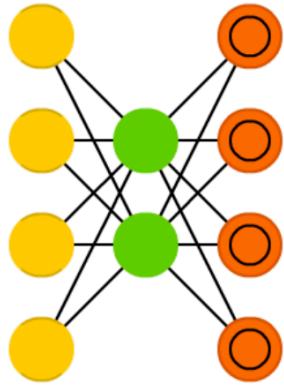
an auto-encoder-decoder is trained to reproduce the input



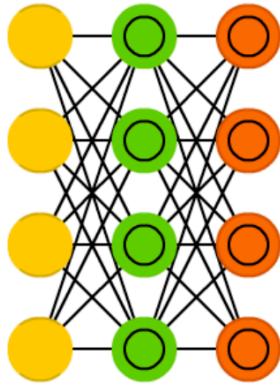
**Reconstruction Loss:** force the ‘hidden layer’ units to become good / reliable feature detectors

# Many Variations

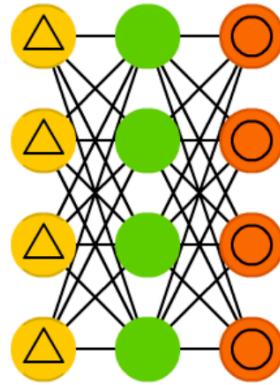
Auto Encoder (AE)



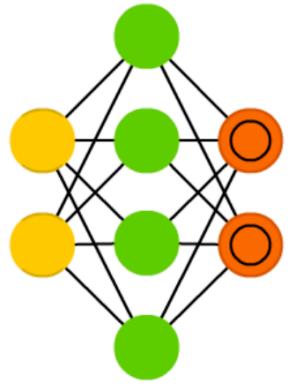
Variational AE (VAE)



Denoising AE (DAE)

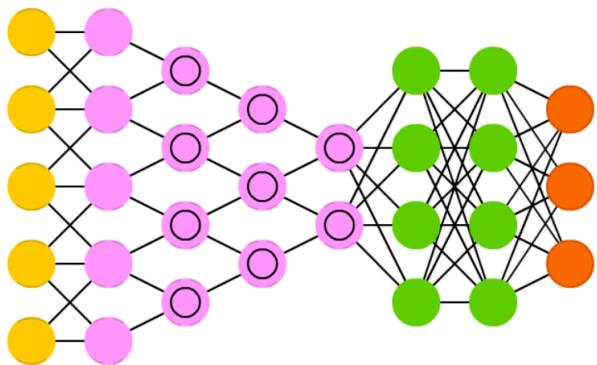


Sparse AE (SAE)

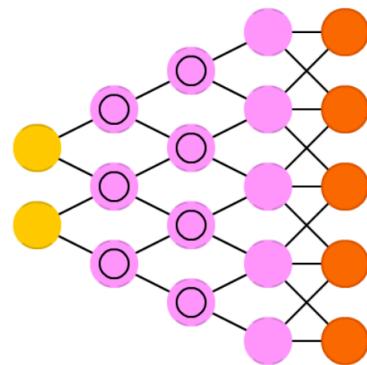


# DCIGN

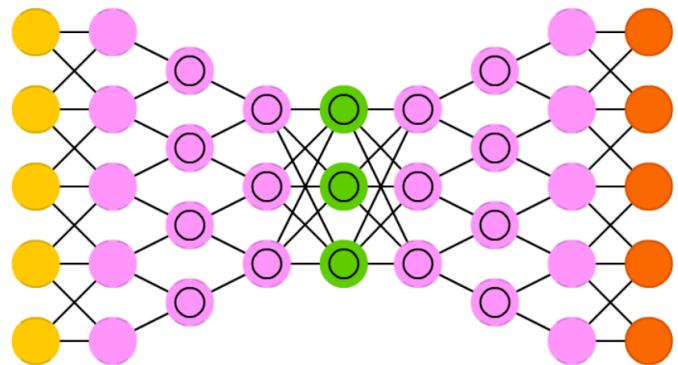
Deep Convolutional Network (DCN)



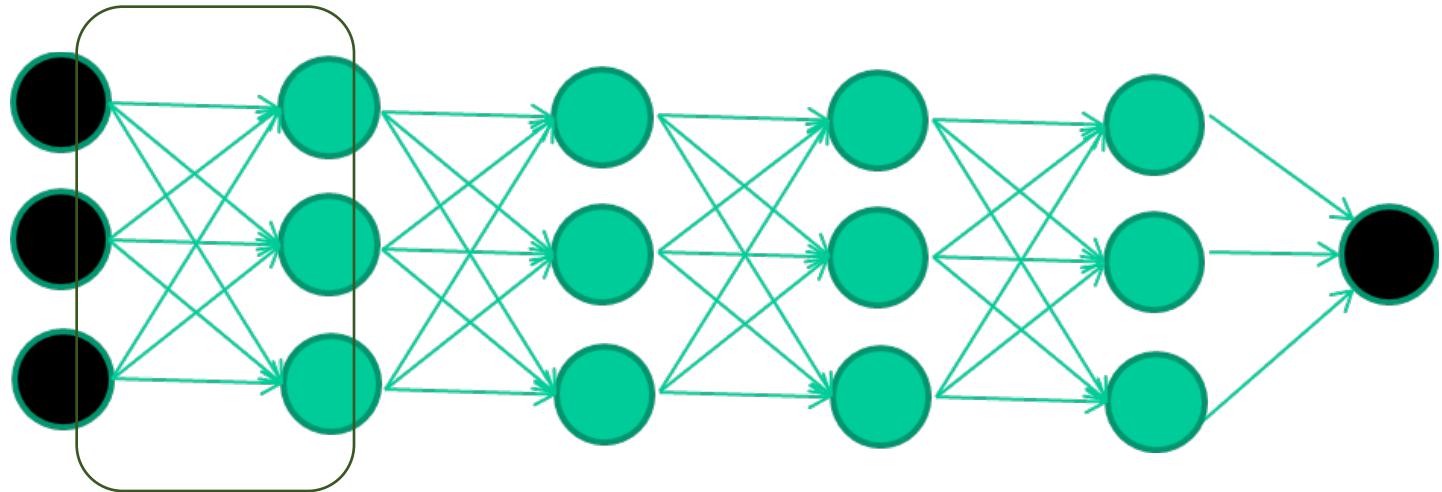
Deconvolutional Network (DN)



Deep Convolutional Inverse Graphics Network (DCIGN)

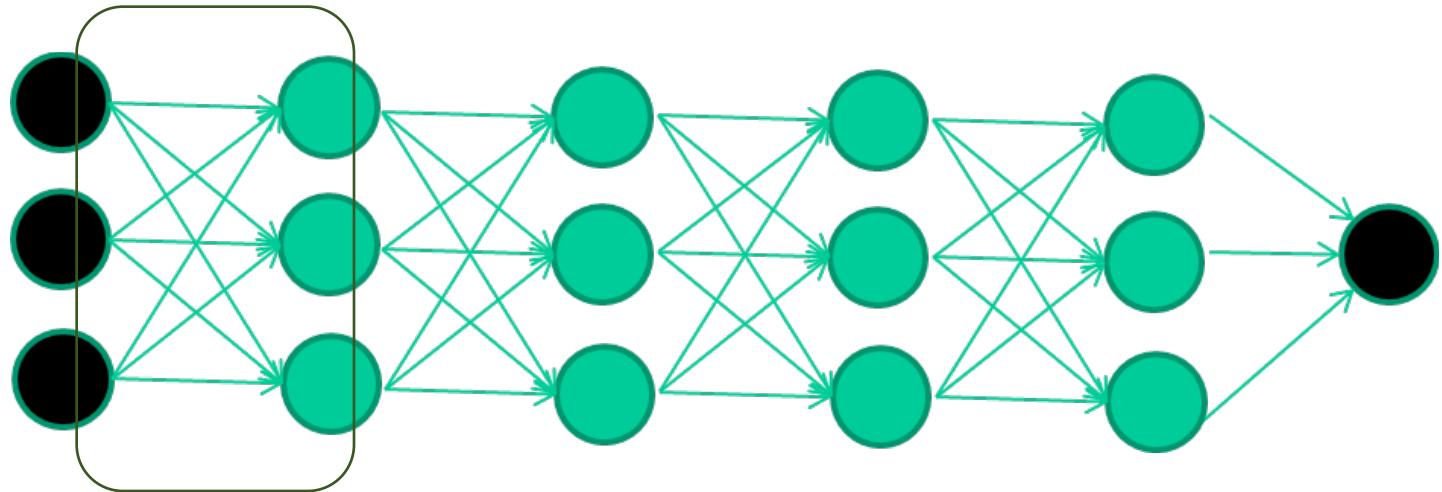


# The new way to train multi-layer NNs...



Train **this** layer first

# The new way to train multi-layer NNs...



*Each layer can be trained well first with some types of self-supervised loss (e.g. reconstruction loss)*

*Basically, it is forced to learn good features that describe what comes from the previous layer*

# Unsupervised Pre-training

- Use our original idea, but **pick a better starting point**
- **Train each level** of the model in a **greedy way**

## 1. Unsupervised Pre-training

- Use **unlabeled** data
- Work bottom-up
  - Train hidden layer 1. Then fix its parameters.
  - Train hidden layer 2. Then fix its parameters.
  - ...
  - Train hidden layer n. Then fix its parameters.

## 2. Supervised Fine-tuning

- Use **labeled** data to train following “Idea #1”
- Refine the features by backpropagation so that they become tuned to the end-task

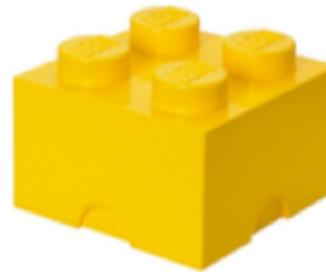
# Pre-Training of DNN layer-layer

- DNN Model Training can be tricky due to...
  - Nonconvexity
  - Vanishing gradients
- Layer-wise pre-training can help with both!
  - Unsupervised
  - Supervised
  - learn features at different levels of abstraction

# Recent Trend (7): Generative Adversarial Networks (GAN)



**Optimization**



**Losses**



MIT  
Technology  
Review

### Dueling Neural Networks

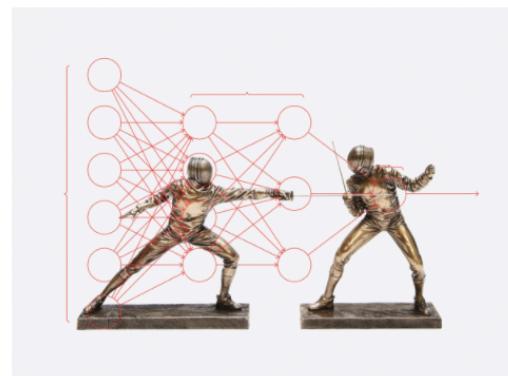
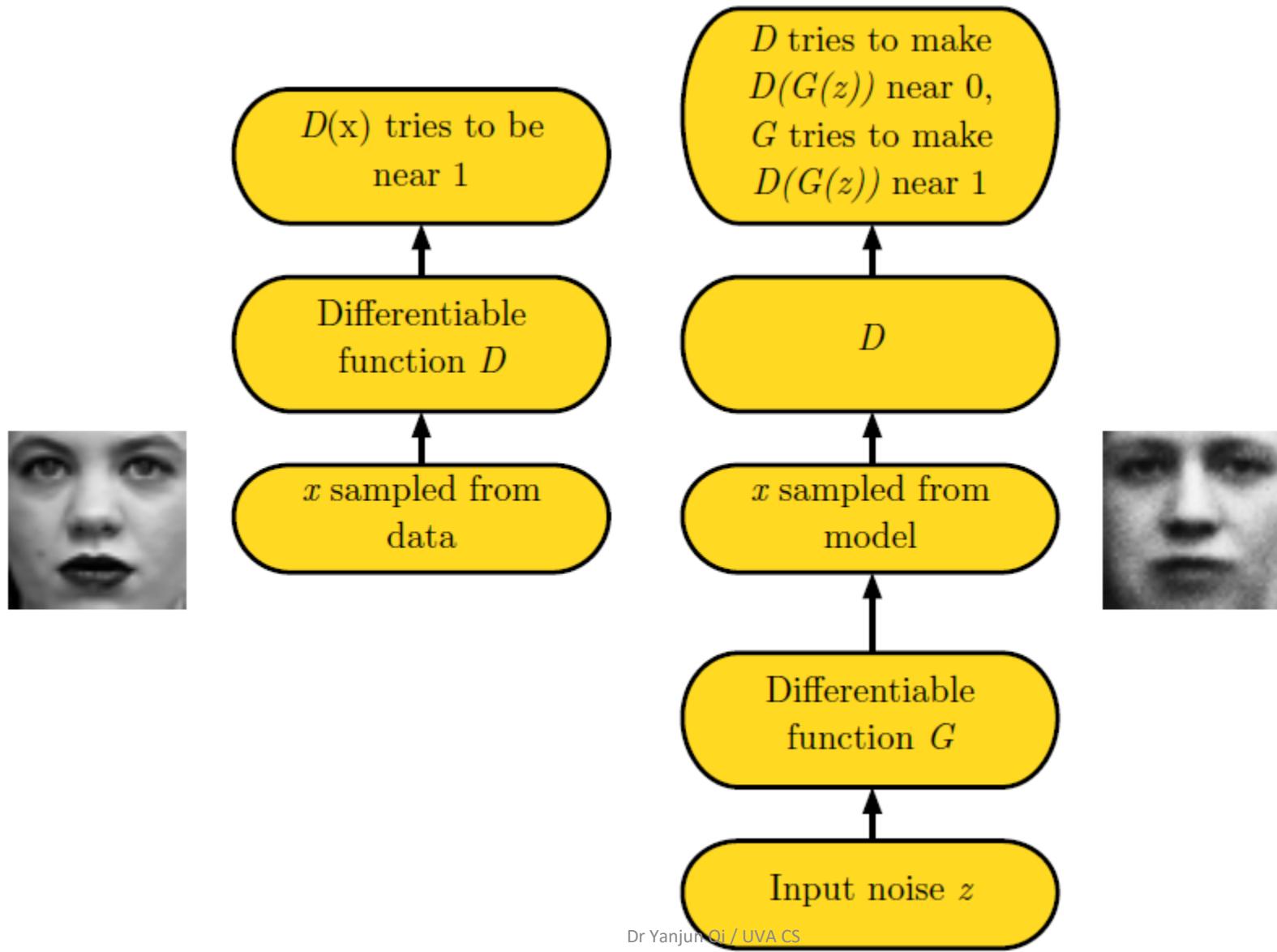


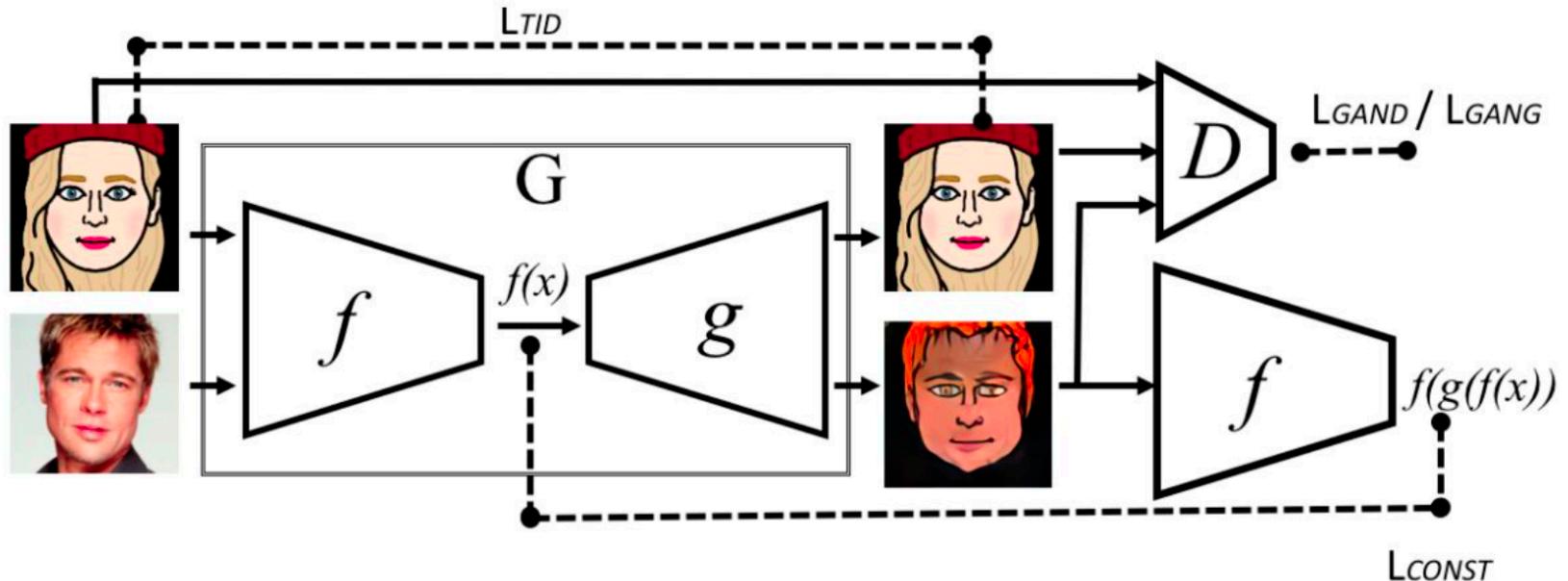
ILLUSTRATION BY DEREK BRAHNEY | DIAGRAM COURTESY OF MICHAEL NIELSEN, "NEURAL NETWORKS AND DEEP LEARNING", DETERMINATION PRESS, 2015

# Adversarial Nets Framework



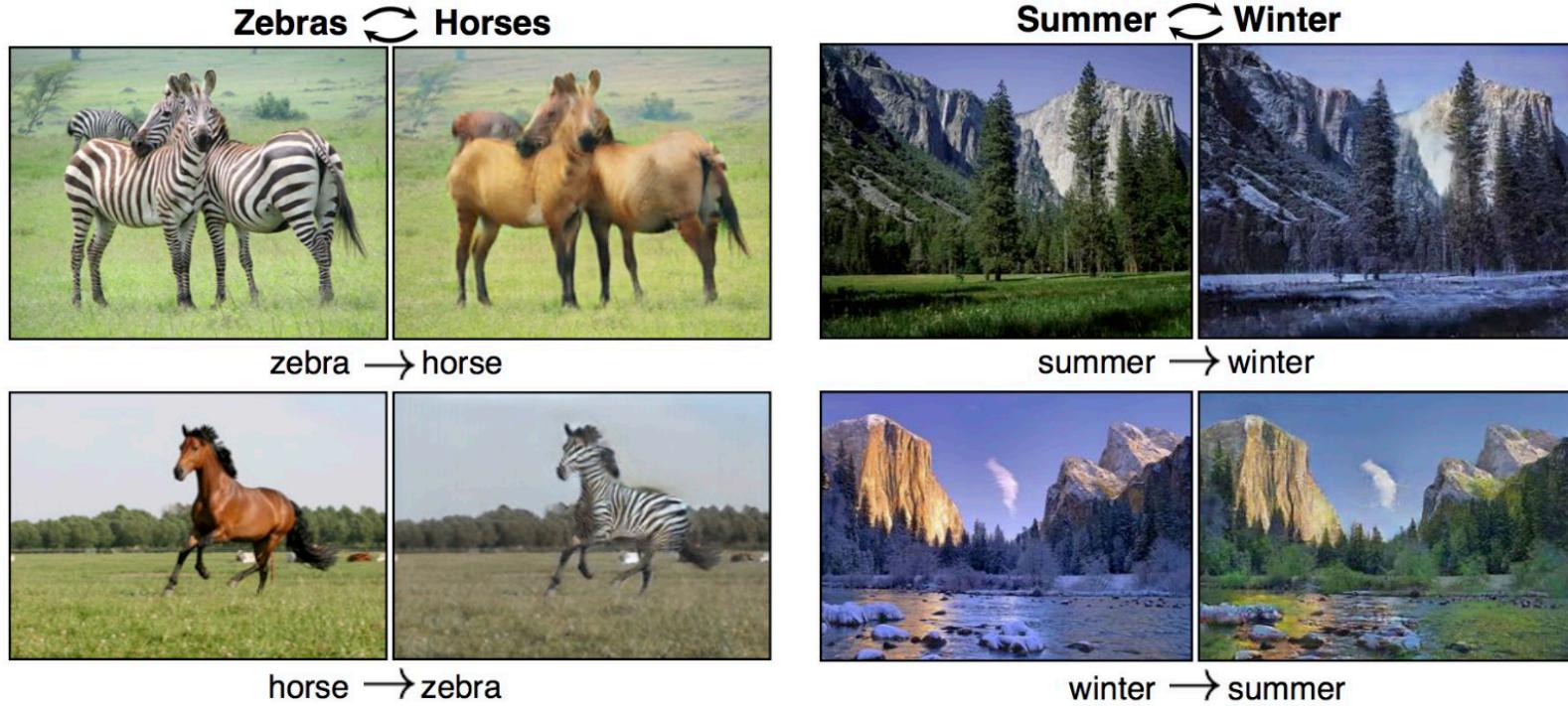


# Unsupervised cross-domain image generation



1. Taigmen et al. "Unsupervised Cross-domain image generation". In *ICLR 2017*.

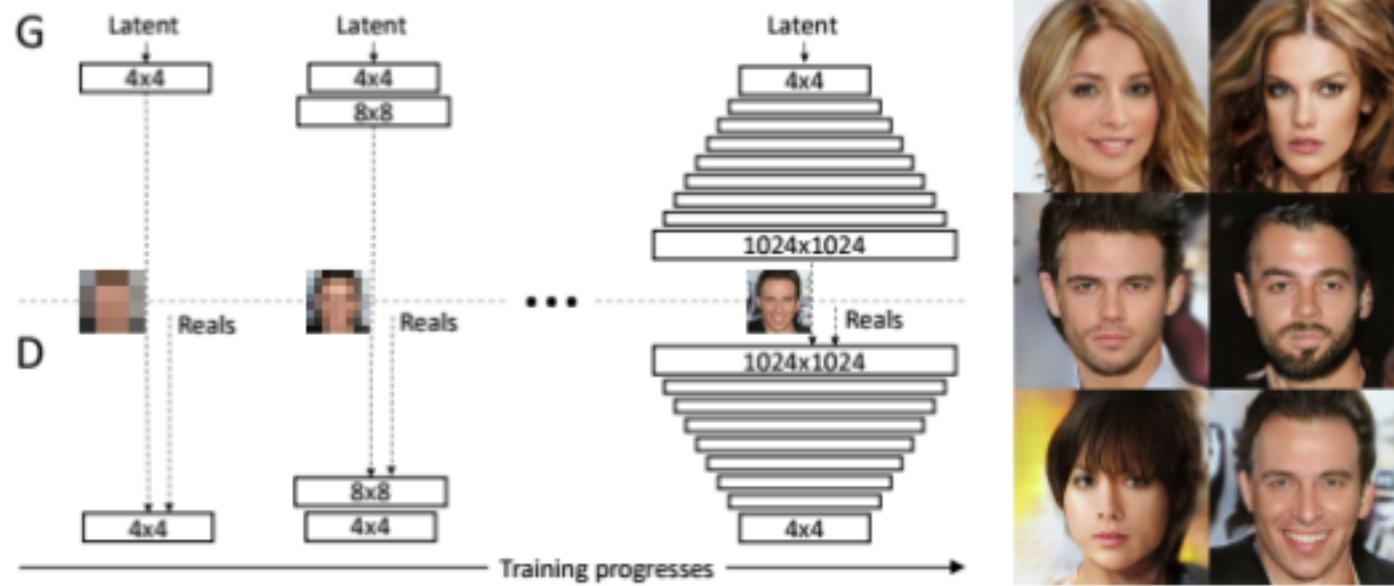
# CycleGAN



1. Zhu et al. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks". In ICCV, 2017.

This paper captures special characteristics of one image collection and figures out how these characteristics could be translated into the other image collection, all in the absence of any paired training examples. CycleGANs method can also be applied in variety of applications such as Collection Style Transfer, Object Transfiguration, season transfer and photo enhancement.

# Progressive GAN



PROGRESSIVE GROWING OF GANS FOR IMPROVED QUALITY, STABILITY, AND VARIATION, ICLR 2018

# A Style-Based Generator Arch Generative Adversarial Ne

Tero Karras  
NVIDIA

tkarras@nvidia.com

Samuli Laine  
NVIDIA

slaine@nvidia.com

CVPR 2019

## StyleGAN

- propose a new generator architecture for GAN which can generate high quality images (called StyleGAN)
- leads to an automatically learned, unsupervised separation of high-level attributes (latent space disentangled)
- propose two new automated metrics for quantifying disentanglement
- propose a high quality dataset of human faces (FFHQ)



# Recent Trend (8): Learning to Optimize / Learning to Search DNN architecture



**Optimization**

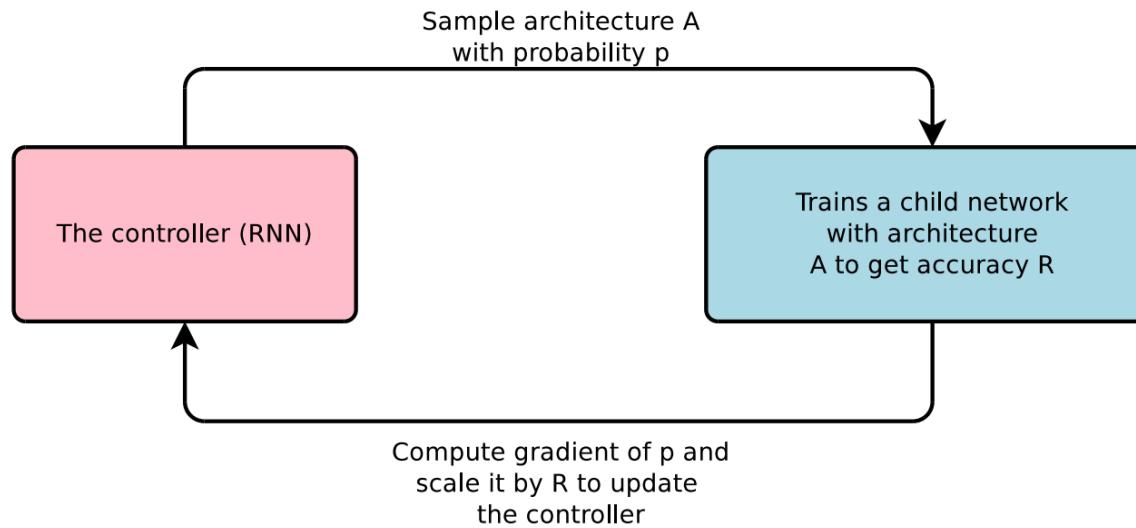
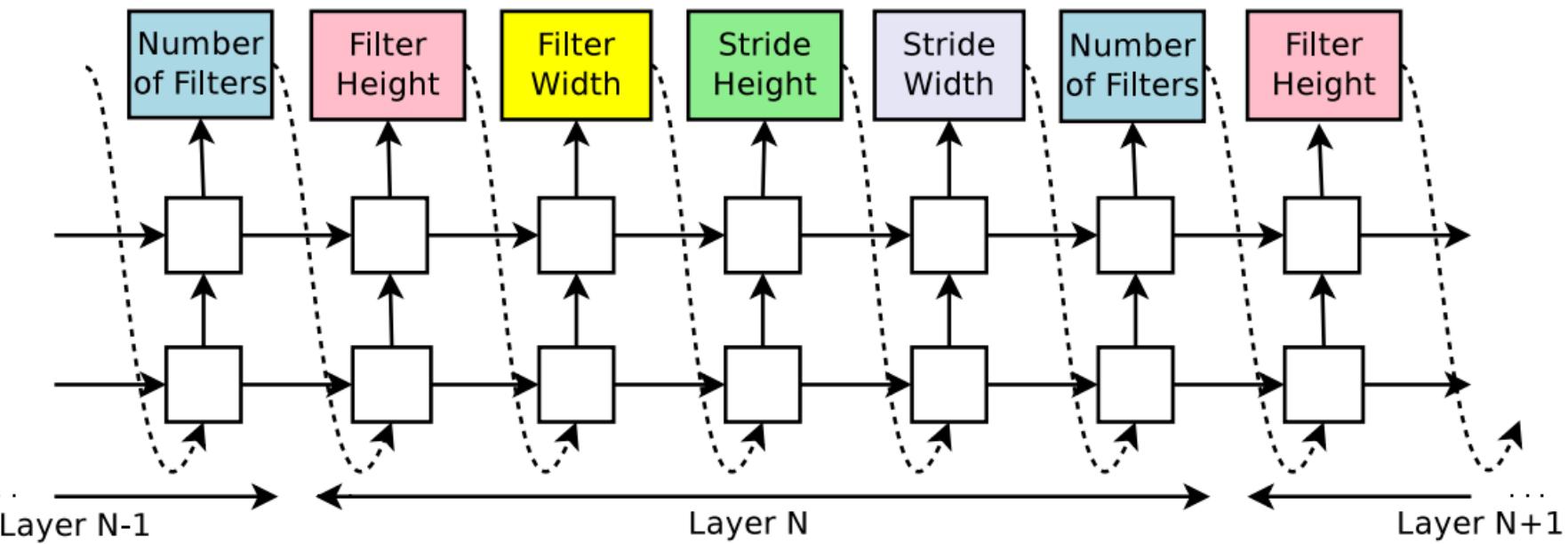


Figure 1: An overview of Neural Architecture Search.



# Neural Optimizer Search with Reinforcement Learning, ICML17

- e.g. hyperpara search

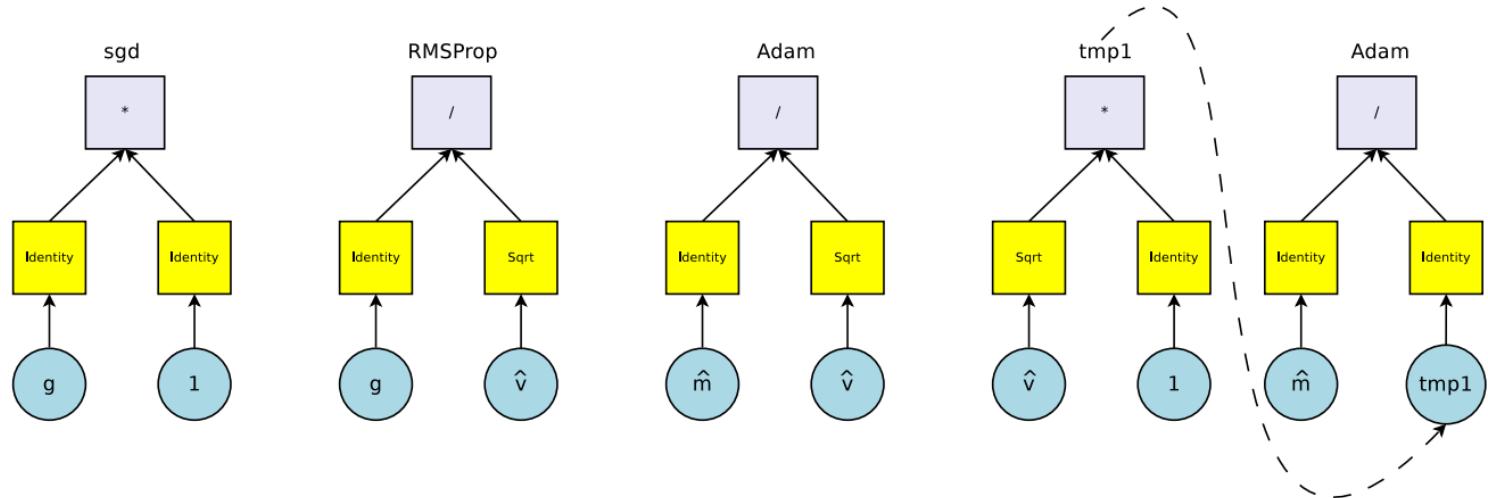


Figure 2. Computation graph of some commonly used optimizers: SGD, RMSProp, Adam. Here, we show the computation of Adam in 1 step and 2 steps. Blue boxes correspond to input primitives or temporary outputs, yellow boxes are unary functions and gray boxes represent binary functions.  $g$  is the gradient,  $\hat{m}$  is the bias-corrected running estimate of the gradient, and  $\hat{v}$  is the bias-corrected running estimate of the squared gradient.

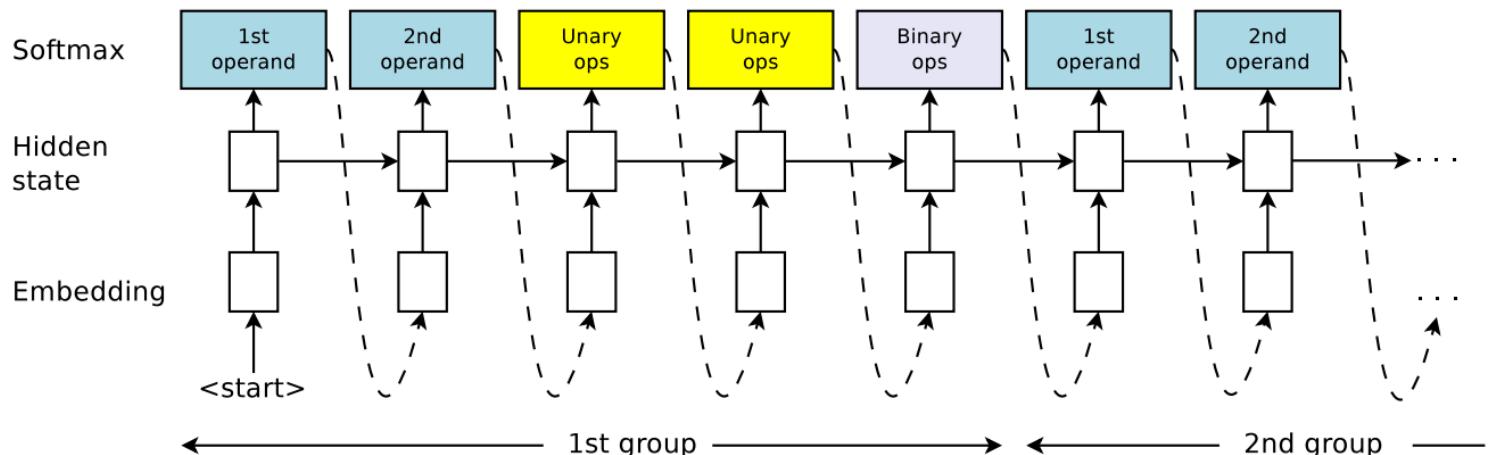


Figure 3. Overview of the controller RNN. The controller iteratively selects subsequences of length 5. It first selects the 1st and 2nd operands  $op_1$  and  $op_2$ , then 2 unary functions  $u_1$  and  $u_2$  to apply to the operands and finally a binary function  $b$  that combines the outputs of the unary functions. The resulting  $b(u_1(op_1), u_2(op_2))$  then becomes an operand that can be selected in the subsequent group of predictions or becomes the update rule. Every prediction is carried out by a softmax classifier and then fed into the next time step as input.

# UVA CS 4774: Machine Learning



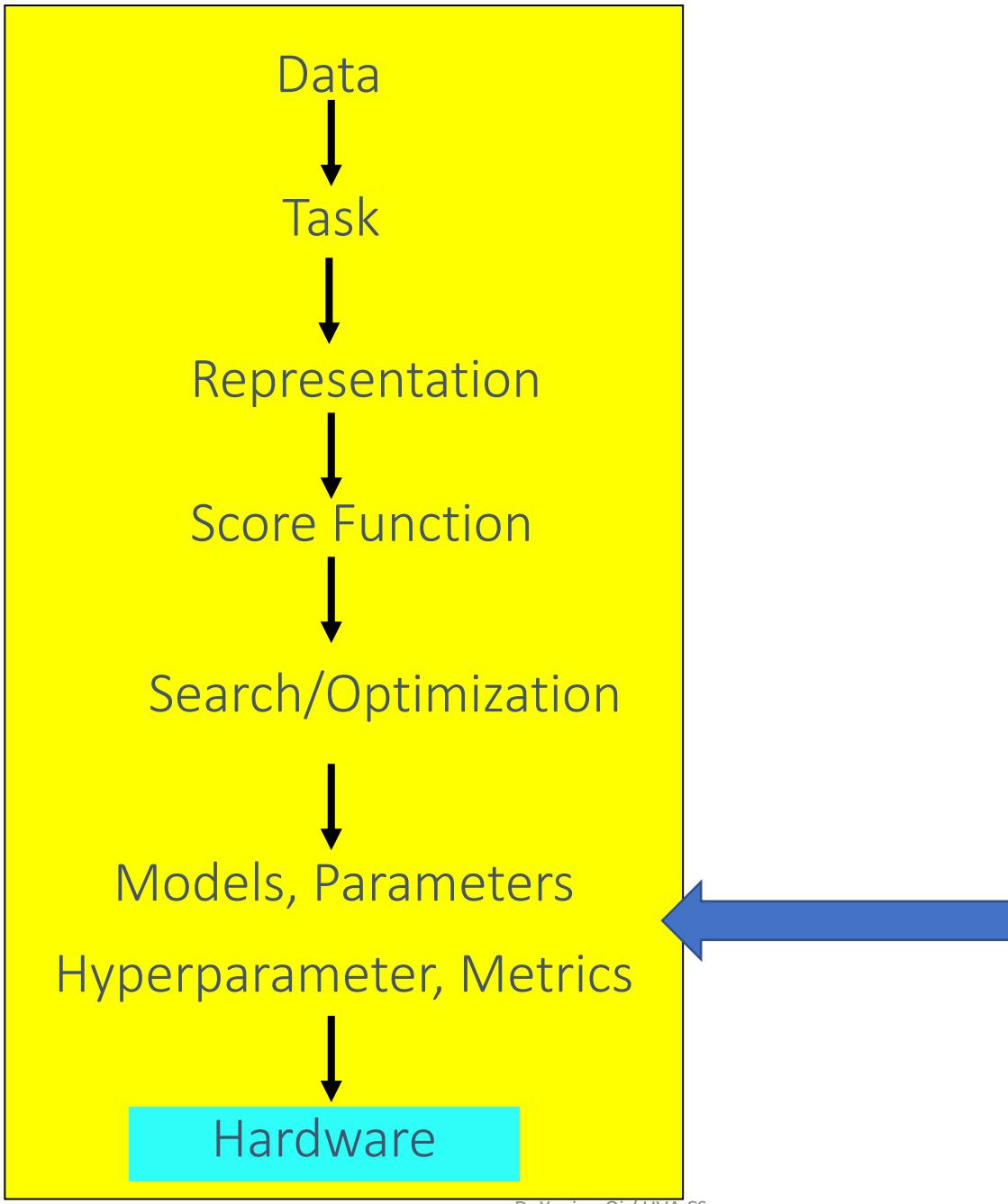
## S3: Lecture 19: Recent Deep Neural Networks: A Quick Overview

Dr. Yanjun Qi

University of Virginia  
Department of Computer Science

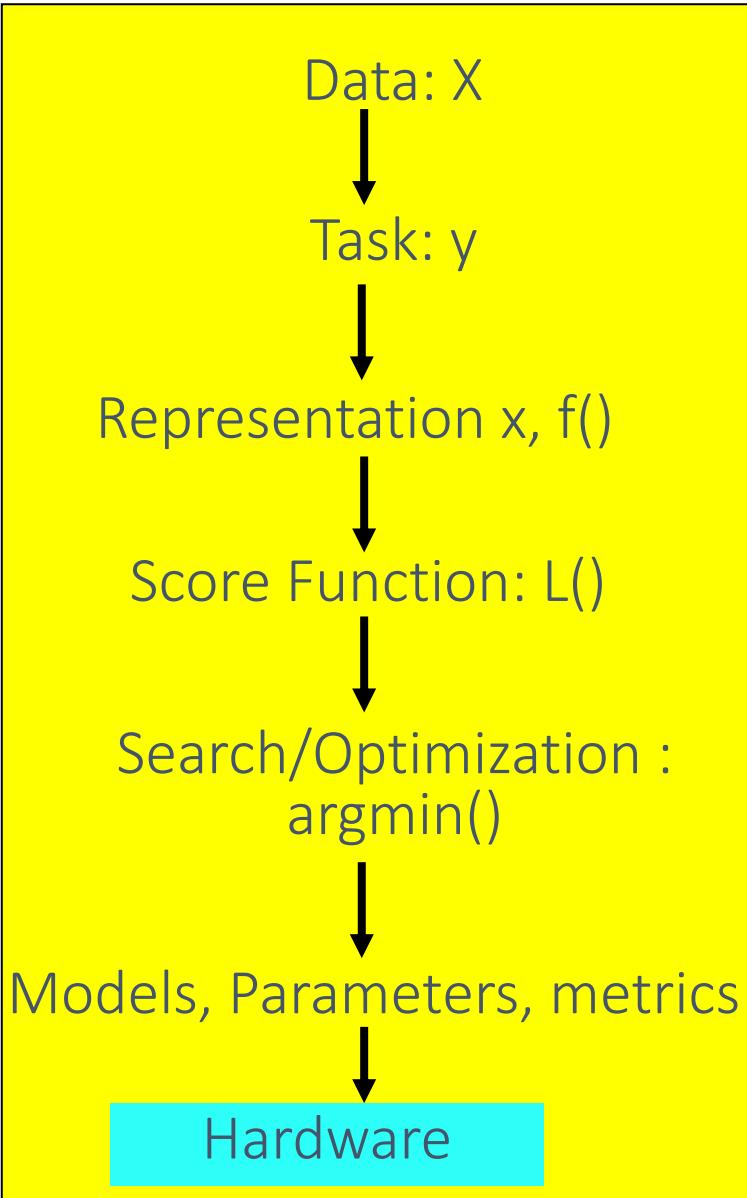
Module IV

# Deep Learning in a Nutshell



# Selected Trends

<https://qdata.github.io/deep2Read/>



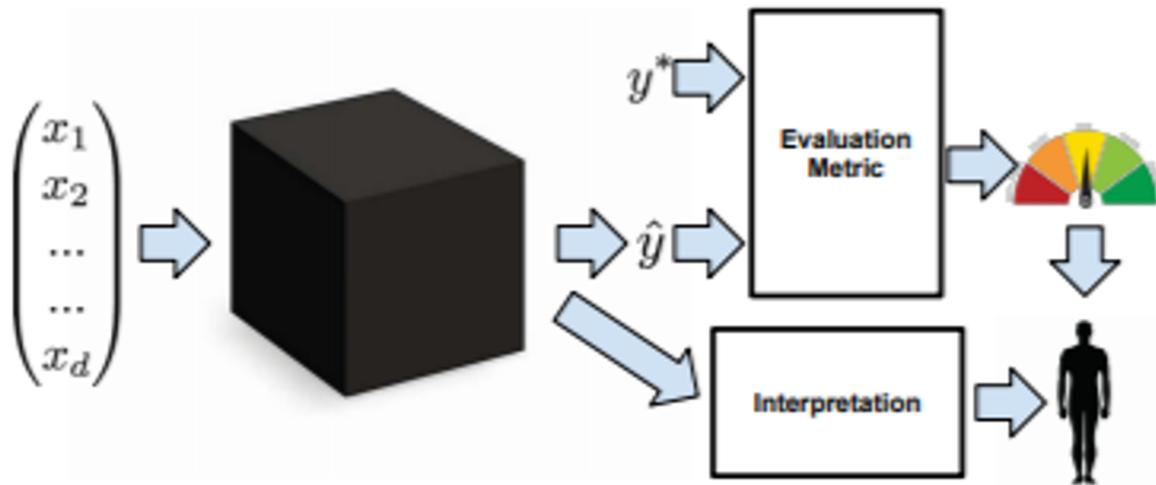
1. Validate / Evade / Test / Verify
2. Understand DNNs

Recent Trend (9):  
Robustness / Trustworthiness / Understand /  
Verify / Test / Evade / Detect Bias / Protect /  
Manage DNN Models



**Validation**

# Understand DNN



**Figure 1.** Typically, evaluation metrics require only predictions and *ground truth* labels. When stakeholders additionally demand *interpretability*, we might infer the existence of desiderata that cannot be captured in this fashion.

- Post-hoc explanations
  - Feature visualization
  - Feature attribution
    - Instance-wise vs. model-wide
  - Feature visualization + attribution
  - Training example attribution
- Inherently interpretable models

# Interpretation Methods: Overview

**Dataset Examples** show us what neurons respond to in practice



**Optimization** isolates the causes of behavior from mere correlations. A neuron may not be detecting what you initially thought.



Baseball—or stripes?  
*mixed4a, Unit 6*

Animal faces—or snouts?  
*mixed4a, Unit 240* D. Yanjia Qi / UVA CS

Clouds—or fluffiness?  
*mixed4a, Unit 453*

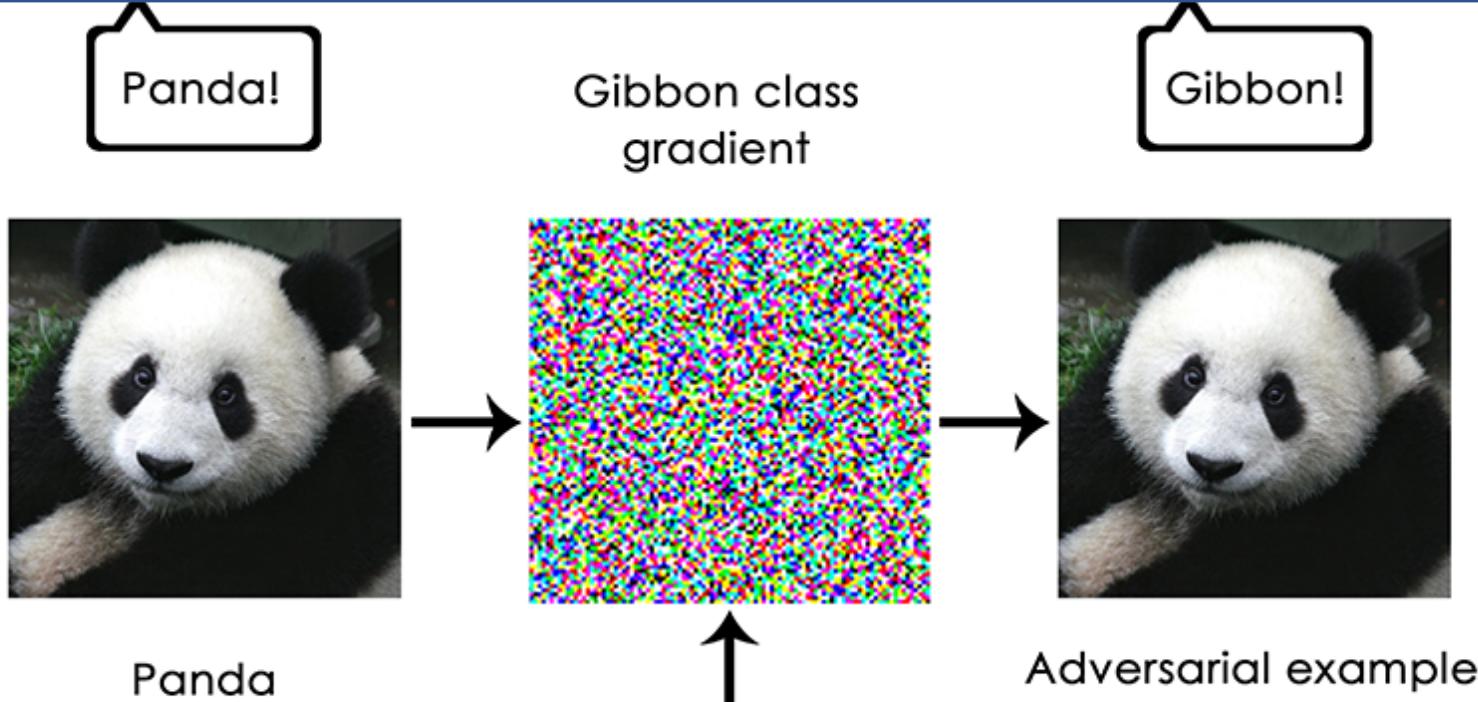
Buildings—or sky?  
*mixed4a, Unit 492*

# To Fool DNN, e.g. Adversarial Examples (AE)

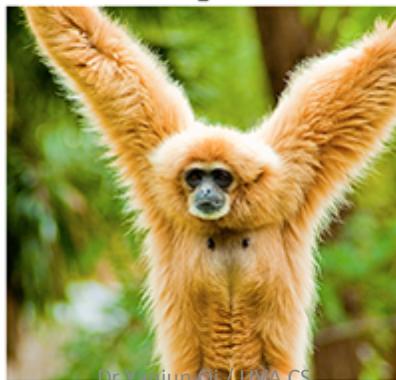
“panda”

$$+ 0.007 \times [noise]$$

“gibbon”

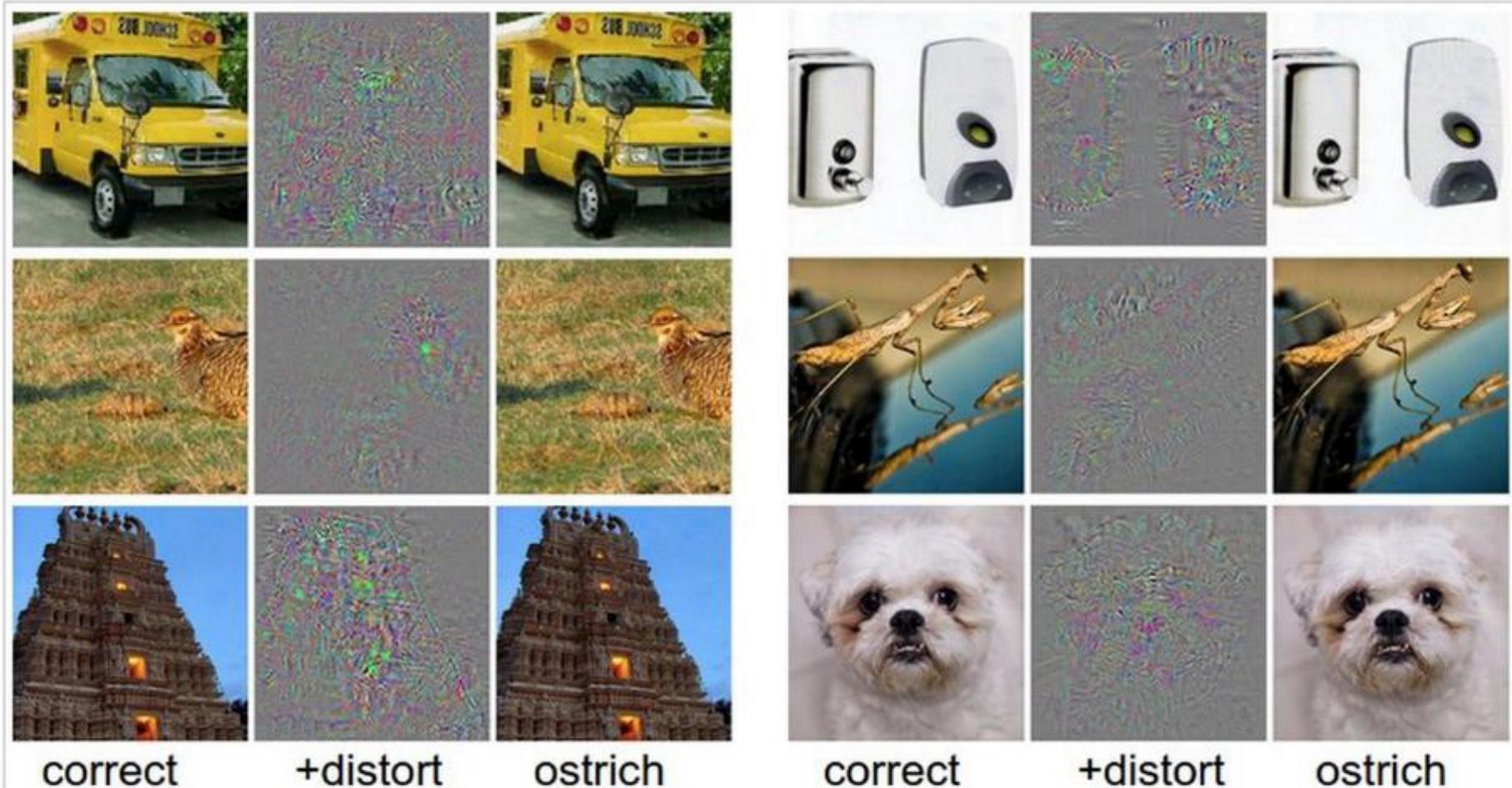


Example from: Ian J.  
Goodfellow, Jonathon  
Shlens, Christian Szegedy.  
Explaining and Harnessing  
Adversarial Examples. ICLR  
2015.



Dr Yanjun Qi / UVA CS

# Breaking CNNs



Take a correctly classified image (left image in both columns), and add a tiny distortion (middle) to fool the ConvNet with the resulting image (right).

Verify DNN, e.g. “Reluplex: An efficient SMT solver for verifying deep neural networks.” International Conference on Computer Aided Verification. 2017.

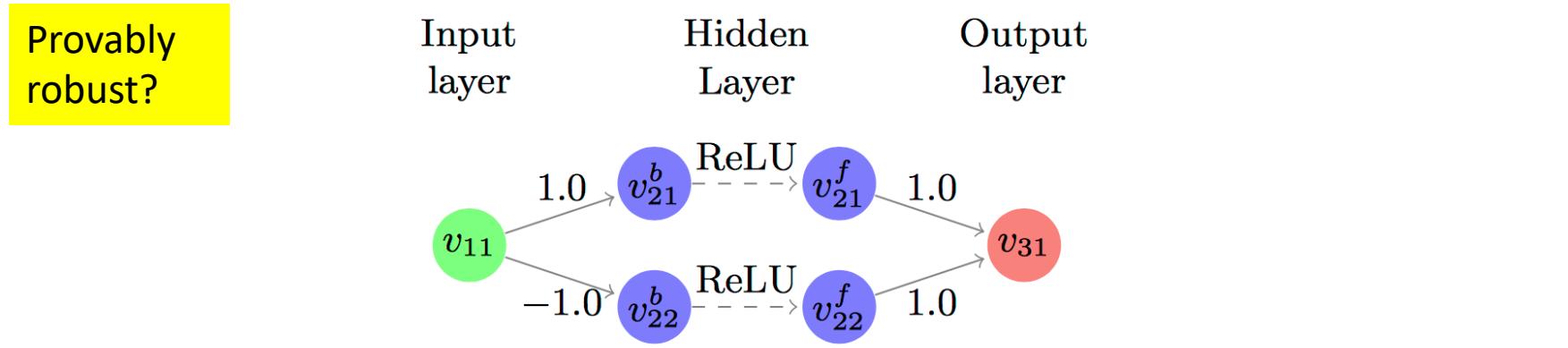
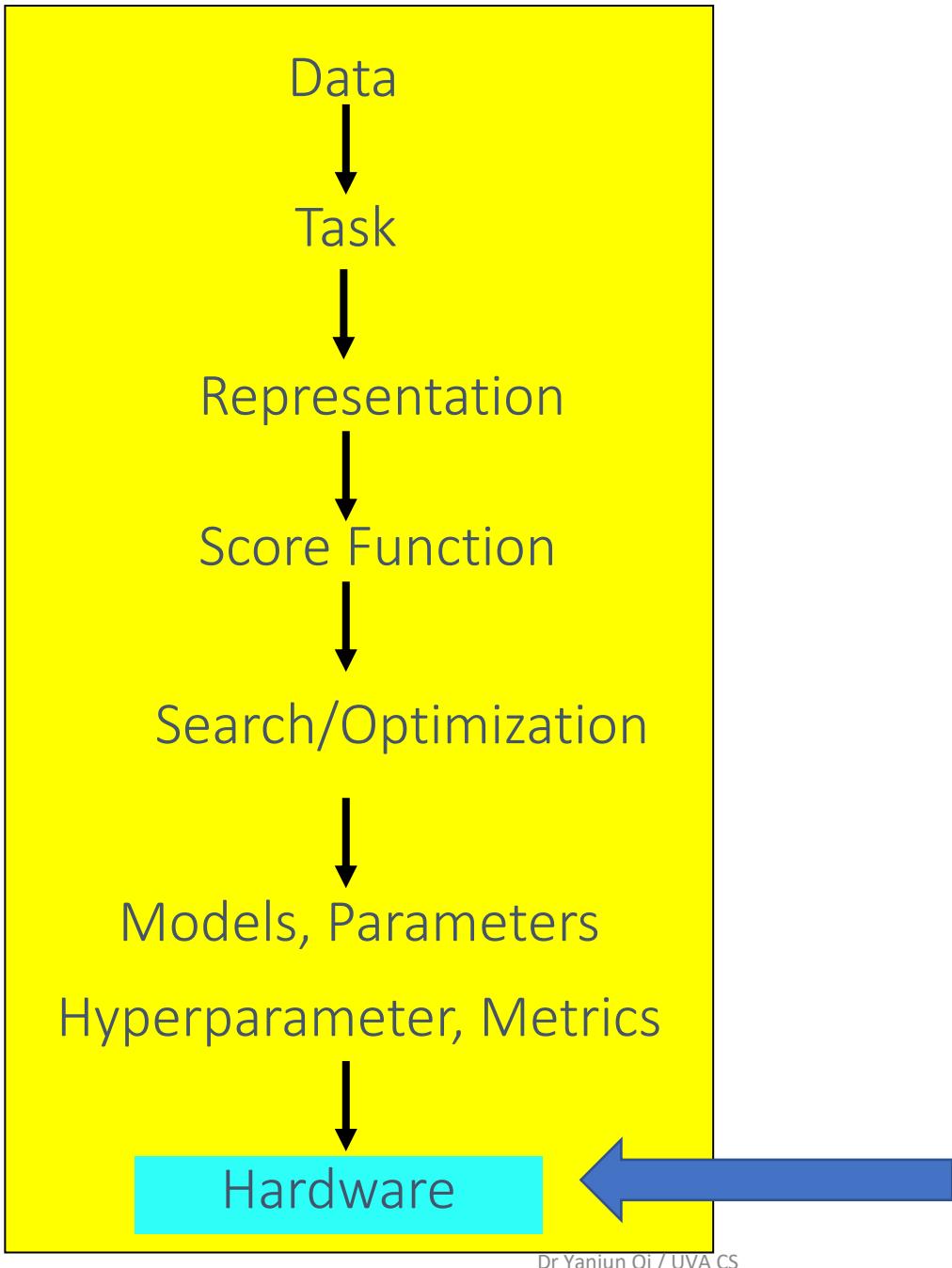


Table 3: Local adversarial robustness tests. All times are in seconds.

	$\delta = 0.1$		$\delta = 0.075$		$\delta = 0.05$		$\delta = 0.025$		$\delta = 0.01$		Total Time
	Result	Time	Result	Time	Result	Time	Result	Time	Result	Time	
Point 1	SAT	135	SAT	239	SAT	24	UNSAT	609	UNSAT	57	1064
Point 2	UNSAT	5880	UNSAT	1167	UNSAT	285	UNSAT	57	UNSAT	5	7394
Point 3	UNSAT	863	UNSAT	436	UNSAT	99	UNSAT	53	UNSAT	1	1452
Point 4	SAT	2	SAT	977	SAT	1168	UNSAT	656	UNSAT	7	2810
Point 5	UNSAT	14560	UNSAT	4344	UNSAT	1331	UNSAT	221	UNSAT	6	20462

# Deep Learning in a Nutshell



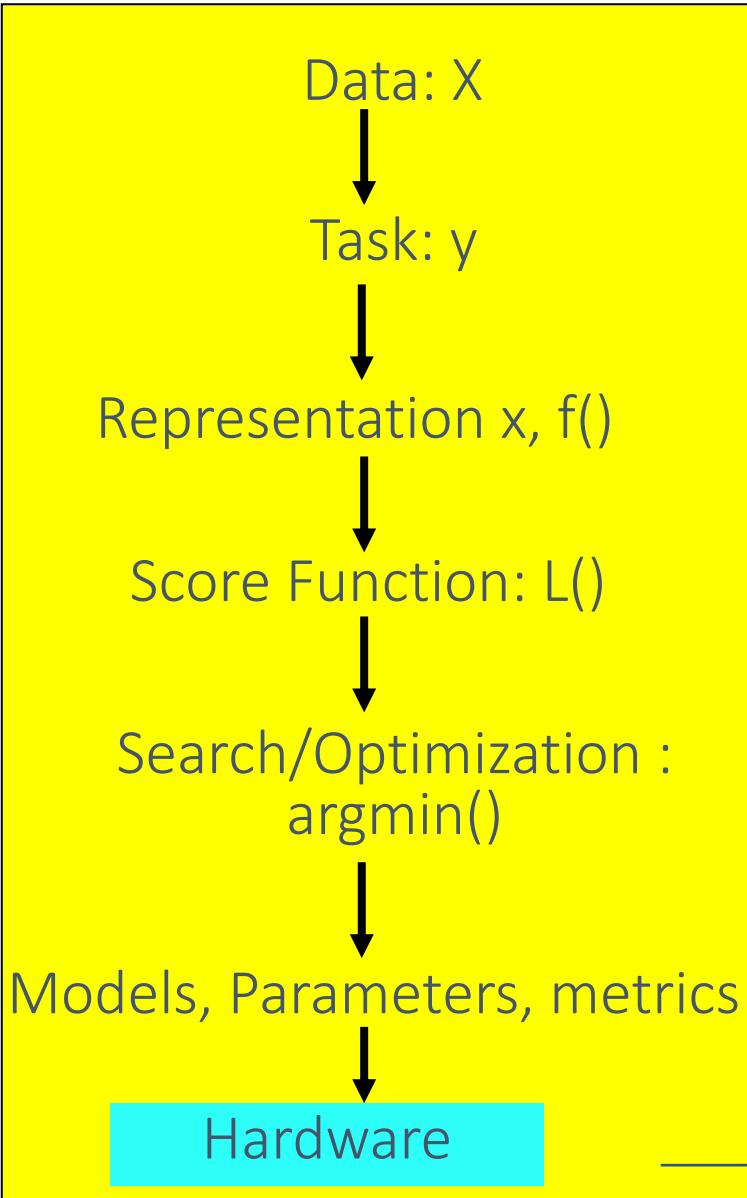
# Recent Trend (10): Hardware and DNN



**Hardware Adaption**

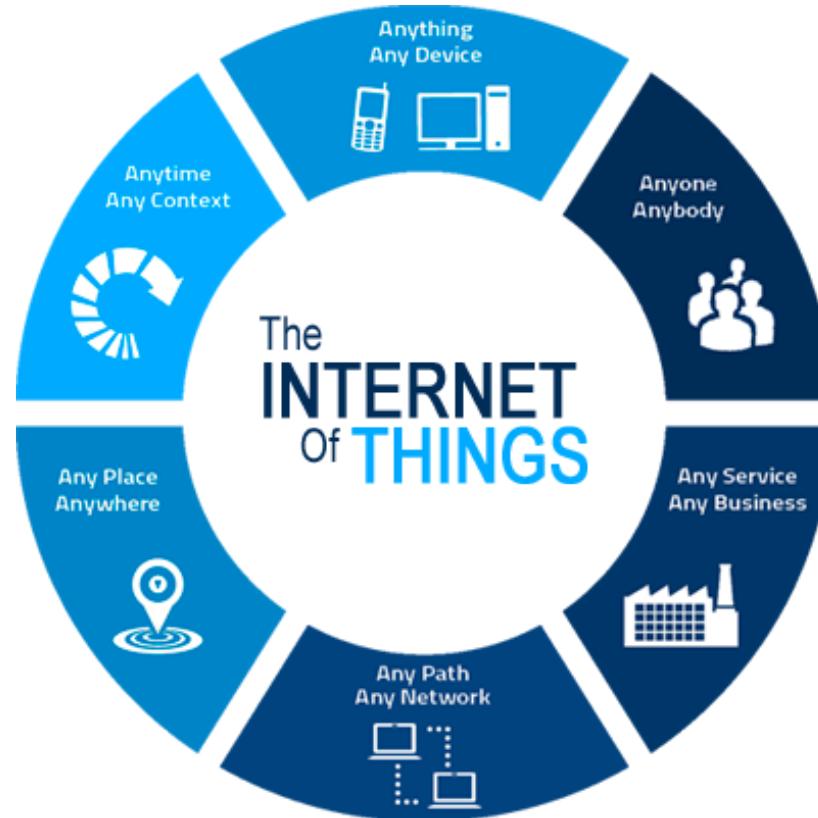
# Selected Trends

<https://qdata.github.io/deep2Read/>



# Applications – efficient edge inference on IoT/ mobile devices

Global IoT spend in  
manufacturing to  
reach \$70 billion by  
2020



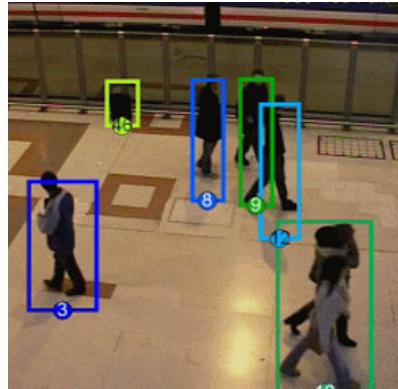
20 billion  
connected devices  
by 2020

# Applications – efficient edge inference on mobile devices



**Battery  
Constrained!**

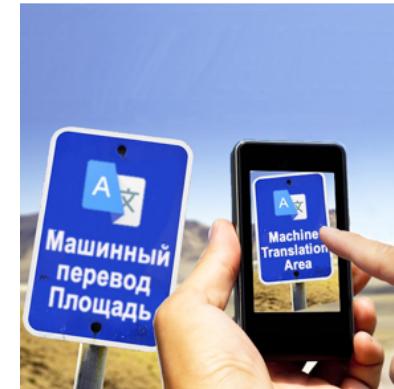
Object detection



Speech recognition



Language translation



Autonomous decision  
making



# DNN Model Compression Methods: Overview

## (a) Model pruning (neurons, filters, kernels, layer)

- Magnitude-based method
  - Iterative pruning + Retraining
  - Pruning with rehabilitation
- Hessian-based method
  - Diagonal Hessian-based method
  - Full Hessian-based method

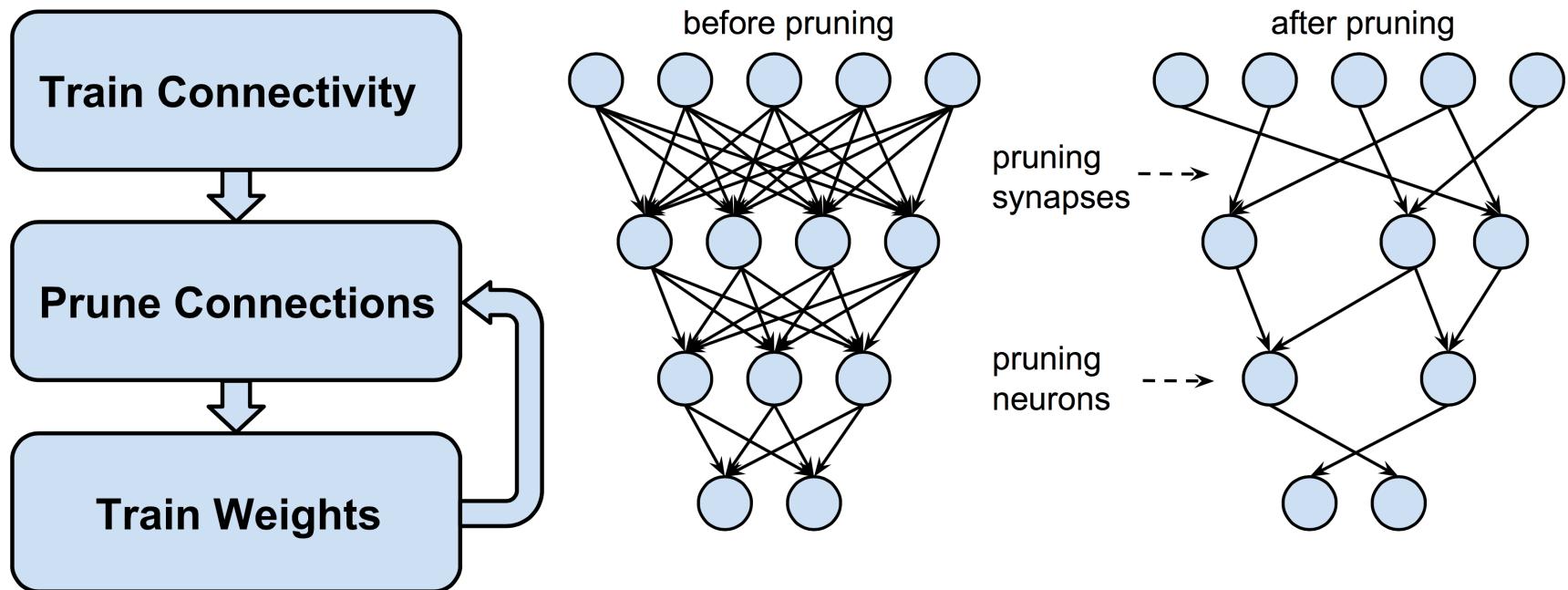
## (b) Construct simpler filters / e.g.

- Decomposition, Matrix Factorization , Singular Value Decomposition (SVD)
- Flattened Convolutions

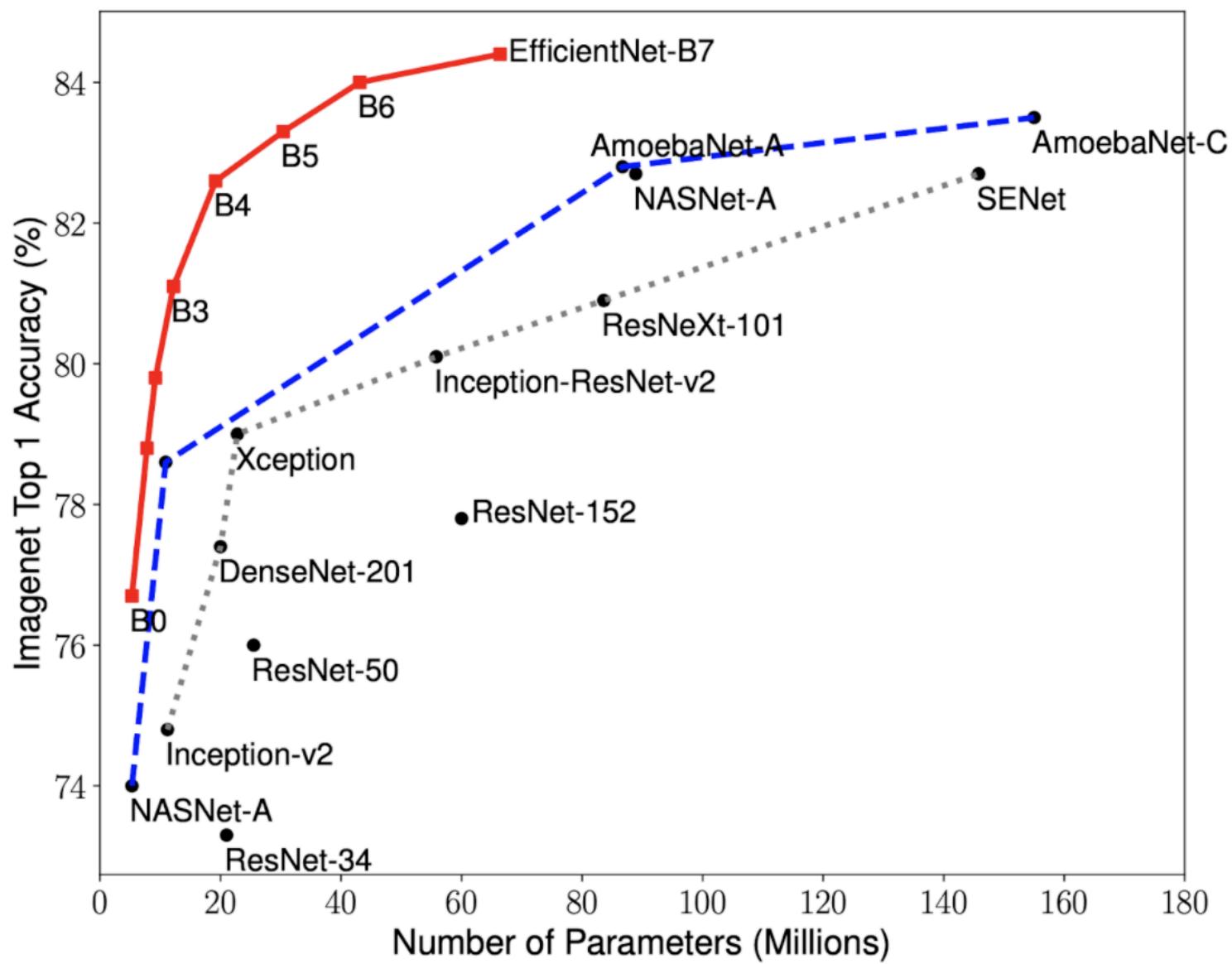
## (c) Quantization of activations, weights, and even gradients.

- Full Quantization
  - Fixed-point format
  - Code book
- Quantization with full-precision copy

For example: Magnitude-based method:  
Iterative Pruning + Retraining

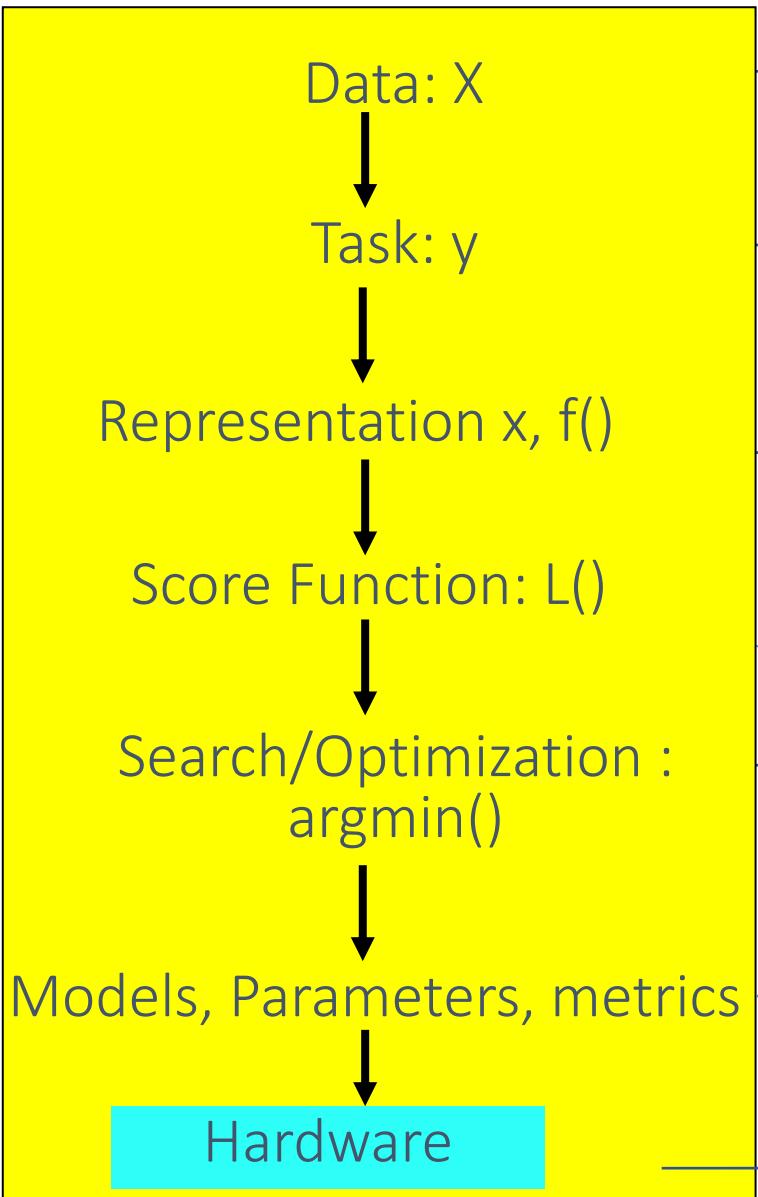


Han, Song, et al. "Learning both weights and connections for efficient neural network." NIPS. 2015.



# Today Recap: Some Selected Trends

<https://qdata.github.io/deep2Read/>



- 1. DNN on graphs / trees / sets
- 2. NTM 4program induction

- 1. Deep Generative models/ DeepFake
- 2. Deep reinforcement learning
- 3. Few-shots / Meta learning / AGI?

- 1. CNN / Residual / Memory
- 2. RNN / Attention / Seq2Seq / Transformer
- 3. Quantitation

- 1. Autoencoder / self supervised training
- 2. Generative Adversarial Networks (GAN)
- 3. Learning to optimize / Learning to search architecture (AutoML)

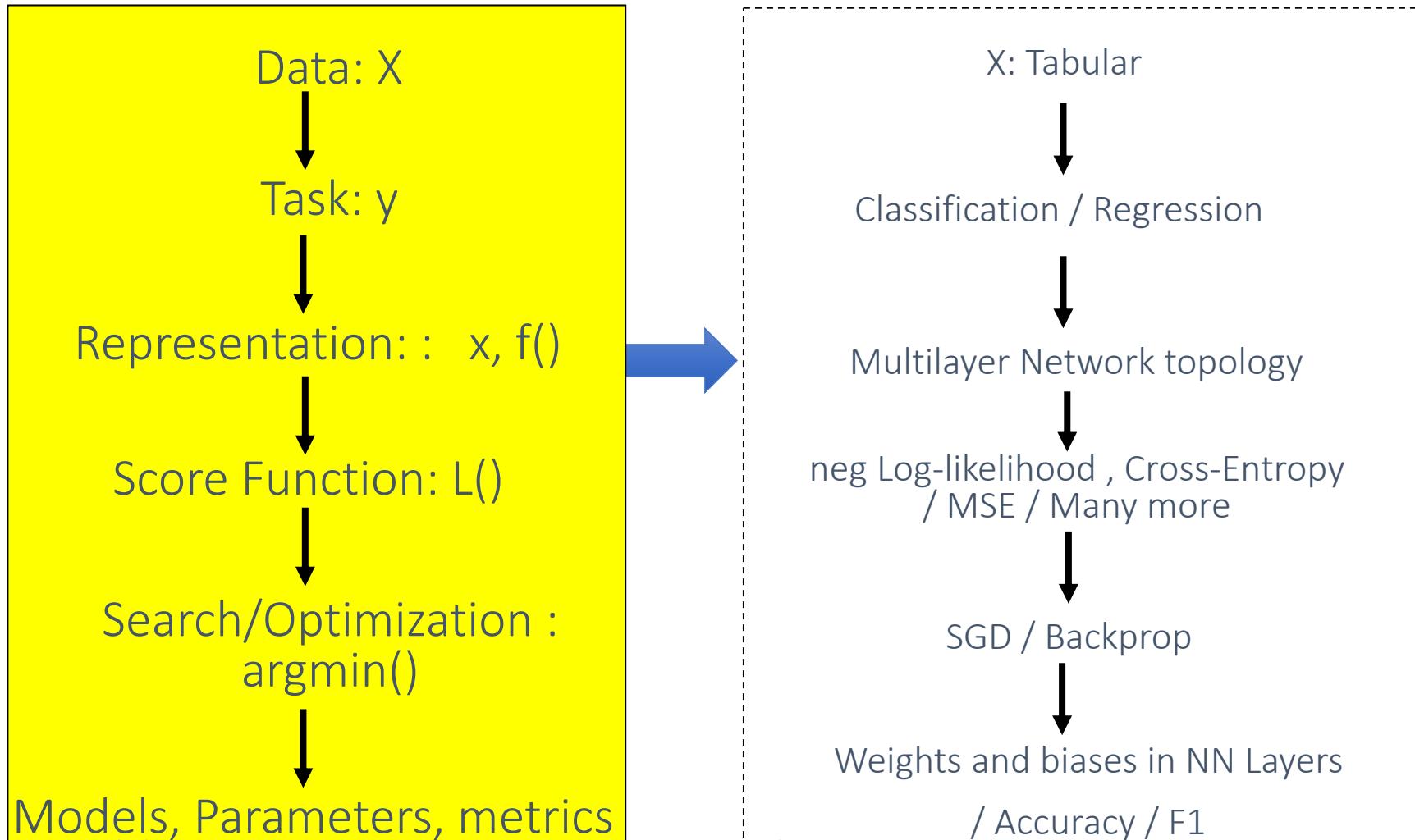
- 1. Validate / Evade / Test / Verify
- 2. Understand DNNs

- 1. Model Compression / Efficient Net

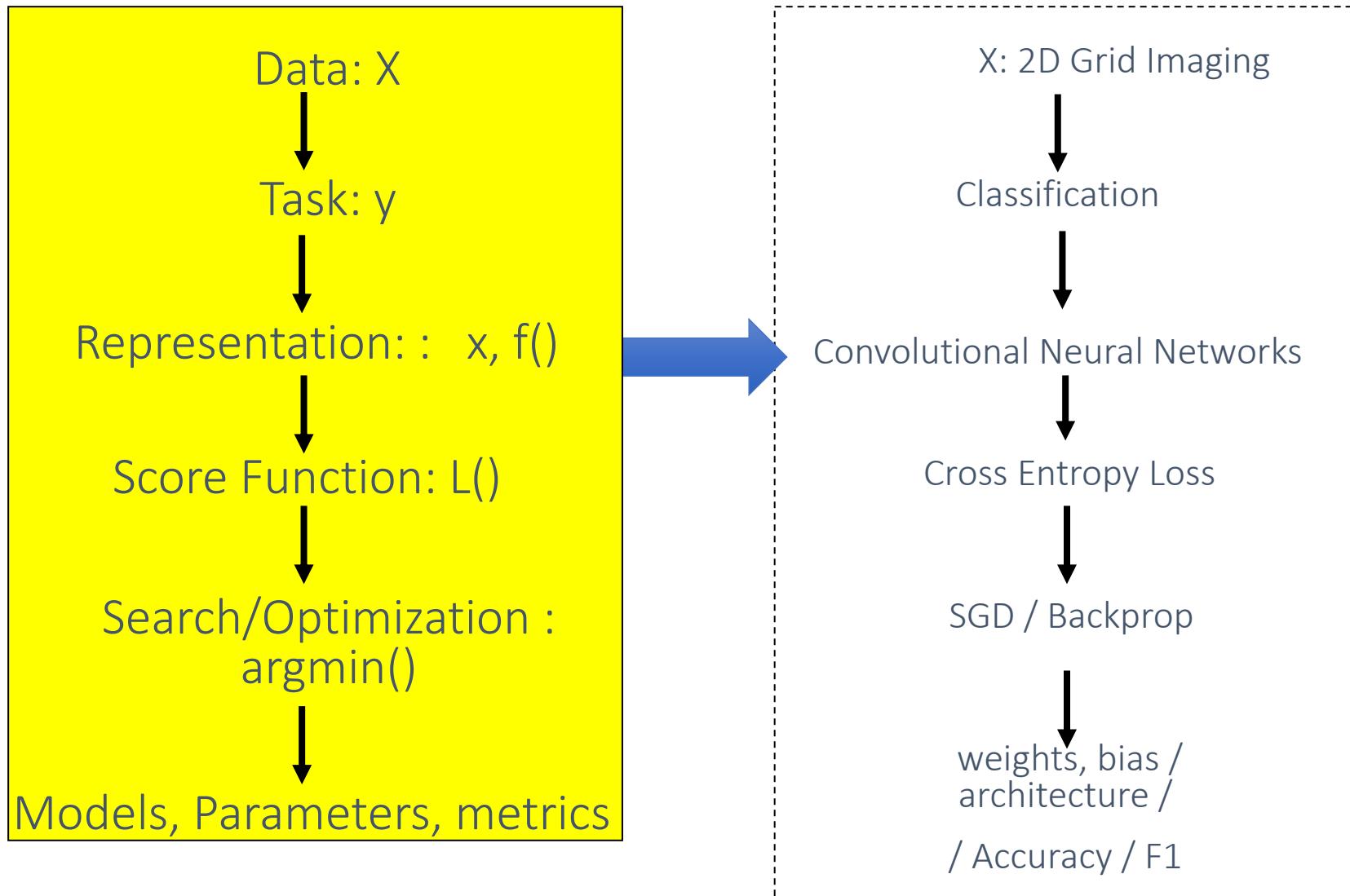
# References

- Dr. Yann Lecun's deep learning tutorials
- Dr. Li Deng's ICML 2014 Deep Learning Tutorial
- Dr. Kai Yu's deep learning tutorial
- Dr. Rob Fergus' deep learning tutorial
- Prof. Nando de Freitas' slides
- Olivier Grisel's talk at Paris Data Geeks / Open World Forum
- Hastie, Trevor, et al. *The elements of statistical learning*. Vol. 2. No. 1. New York: Springer, 2009.
- Dr. Hung-yi Lee's CNN slides
- NIPS 2017 DL Trend Tutorial

# Recap: Basic MLP Neural Network Models



## Recap: Convolutional Network Models on 2D Grid / Image



## Recap: Auto Encoder

