# Deep Neural Networks: A Quick and Rough Overview

Dr. Yanjun Qi

@ UVA Human Machine Intelligence Seminar

# Summary: Some Recent Trends

- 1. Autoencoder / layer-wise training
- 2. CNN / Residual / Dynamic parameter
- 3. RNN / Attention / Seq2Seq, …
- 4. Neural Architecture with explicit memory
- 5. NTM 4program induction / sequential decisions
- 6. Learning to optimize / Learning DNN architectures
- 7. Learning to learn / meta-learning/ few-shots
- 8. DNN on graphs / trees / sets
- 9. Deep Generative models, e.g., autoregressive
- 10. Generative Adversarial Networks (GAN)
- 11. Deep reinforcement learning
- 12. Validate / Evade / Test / Understand / Verify DNNs

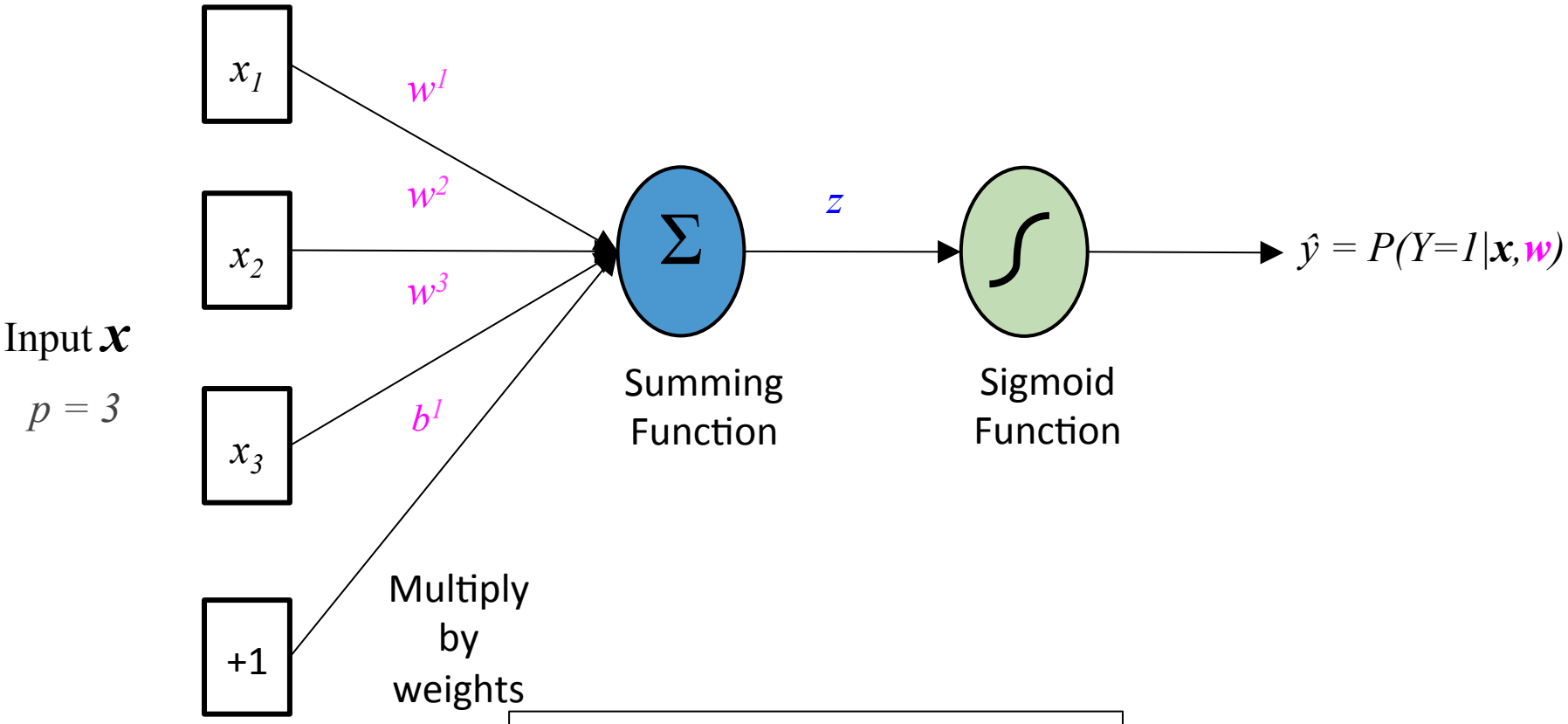
# Today

- Deep Learning
  - Basics
  - History
  - Why is this breakthrough ?
  - Recent trends

# Basics

- Basic Neural Network (NN)
  - single neuron, e.g. logistic regression unit
  - multilayer perceptron (MLP)
  - for multi-class classification, softmax layer
  - training NN with backprob algorithm

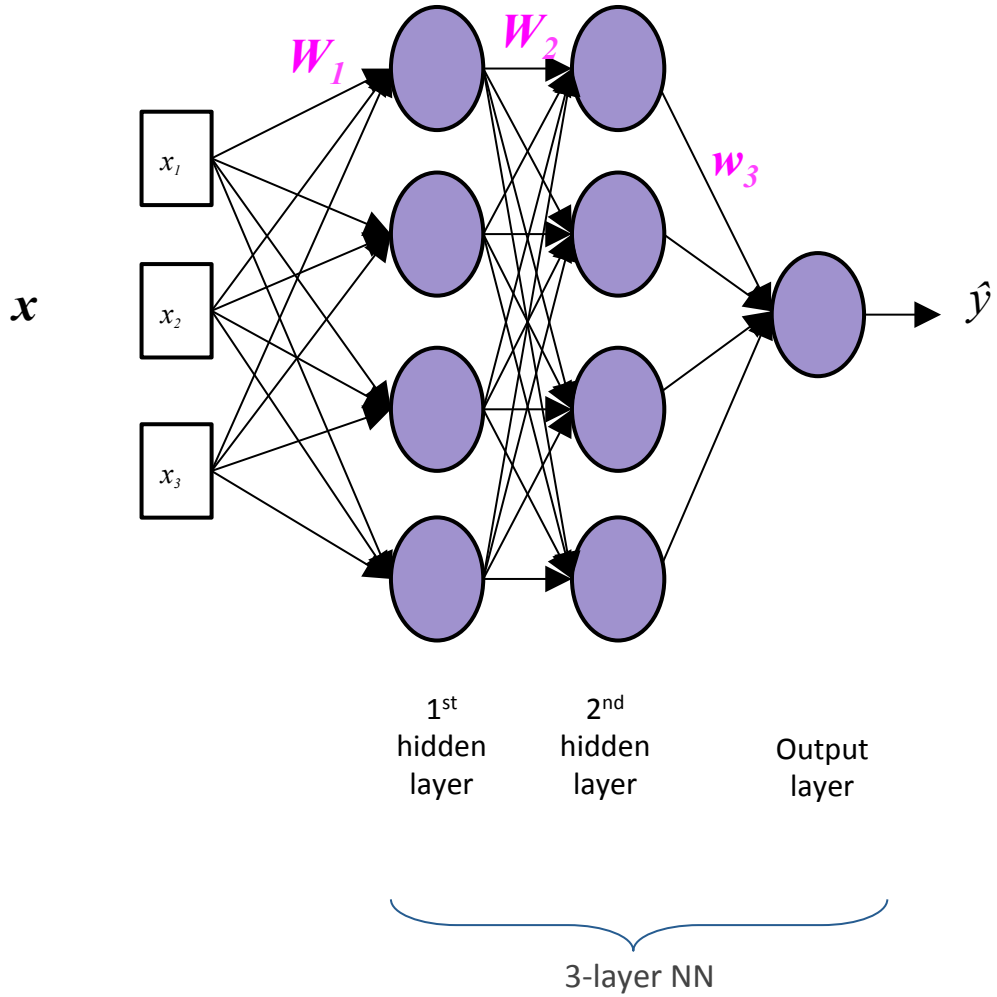
# One "Neuron": Expanded Logistic Regression



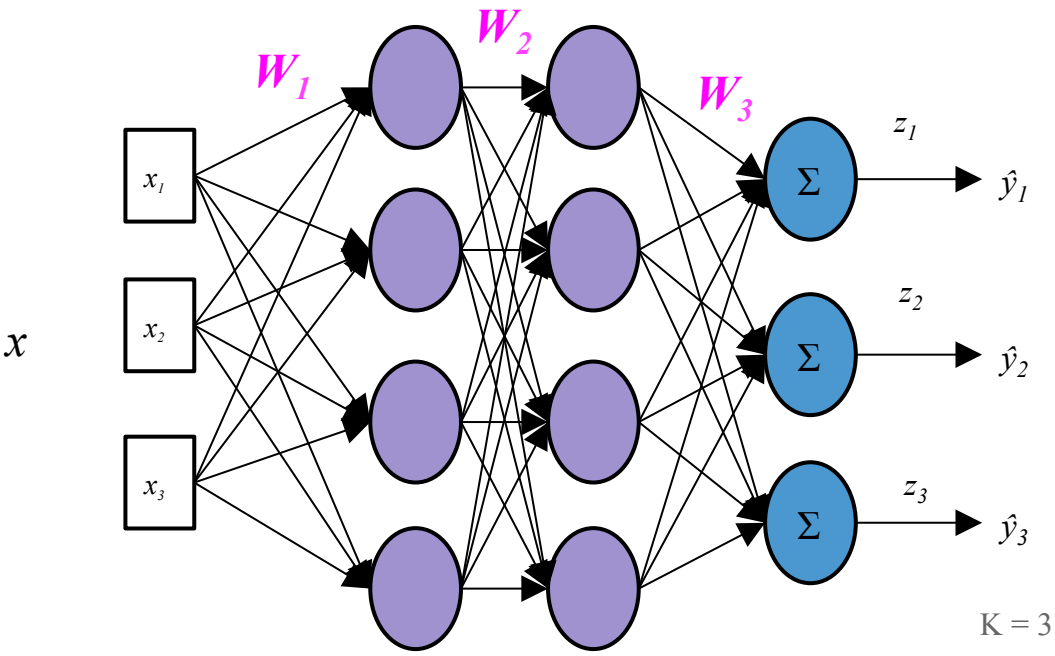

$$z = \boldsymbol{w}^T \cdot \boldsymbol{x} + \boldsymbol{b}$$

$$y = sigmoid(z) = \frac{e^z}{1 + e^z}$$

# Multi-Layer Neural Network (MLP)



# Multi-Class Classification Loss



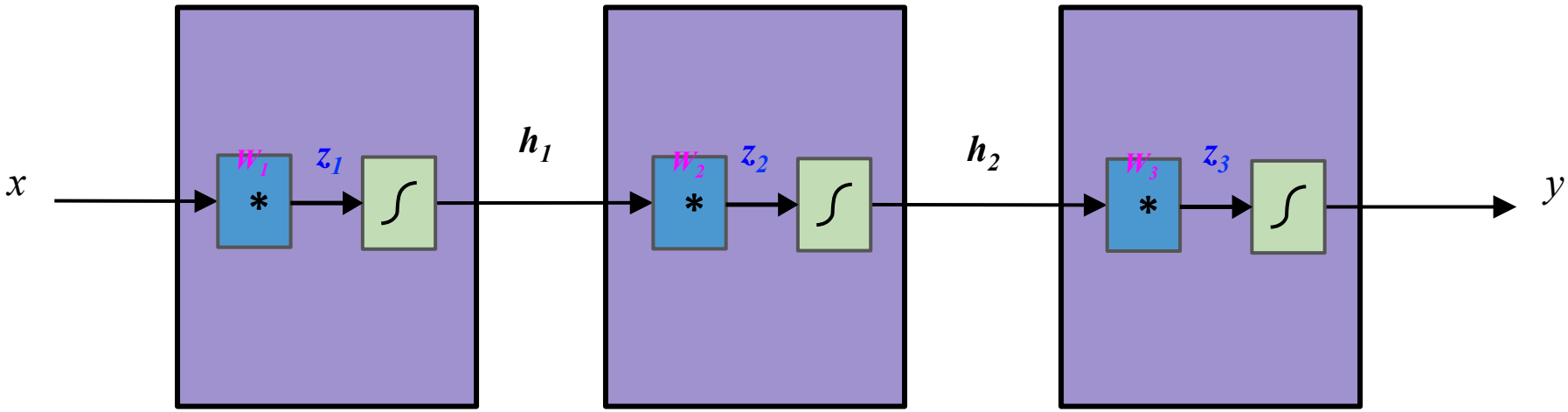

$$\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}} = P(\hat{y}_i = 1 \mid \boldsymbol{x})$$

**"Softmax" function.**
Normalizing function which converts each class output to a probability.

$$E = \text{loss} = -\sum_{j=1\ldots K} y_j \ln \hat{y}_j$$

"0" for all except true class

# “Block View” Of MLP




1st hidden layer

2nd hidden layer

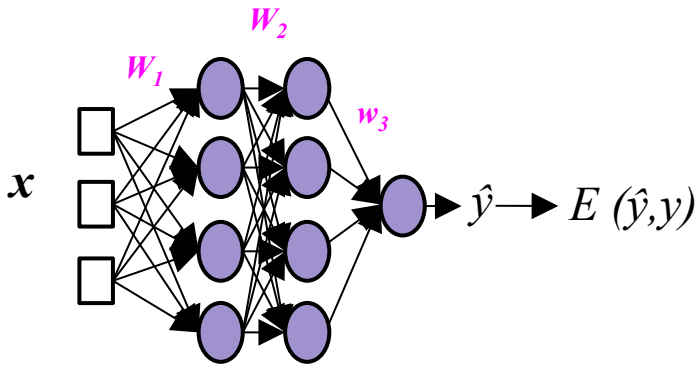
Output layer

# Training Neural Networks

How do we learn the optimal weights $W_L$ for our task??

- **Stochastic Gradient descent:**

$$W_L(x_{t+1}) = W_L(x_t) - \eta \frac{\partial E}{\partial W_L(x_t)}$$




But how do we get gradients of lower layers?

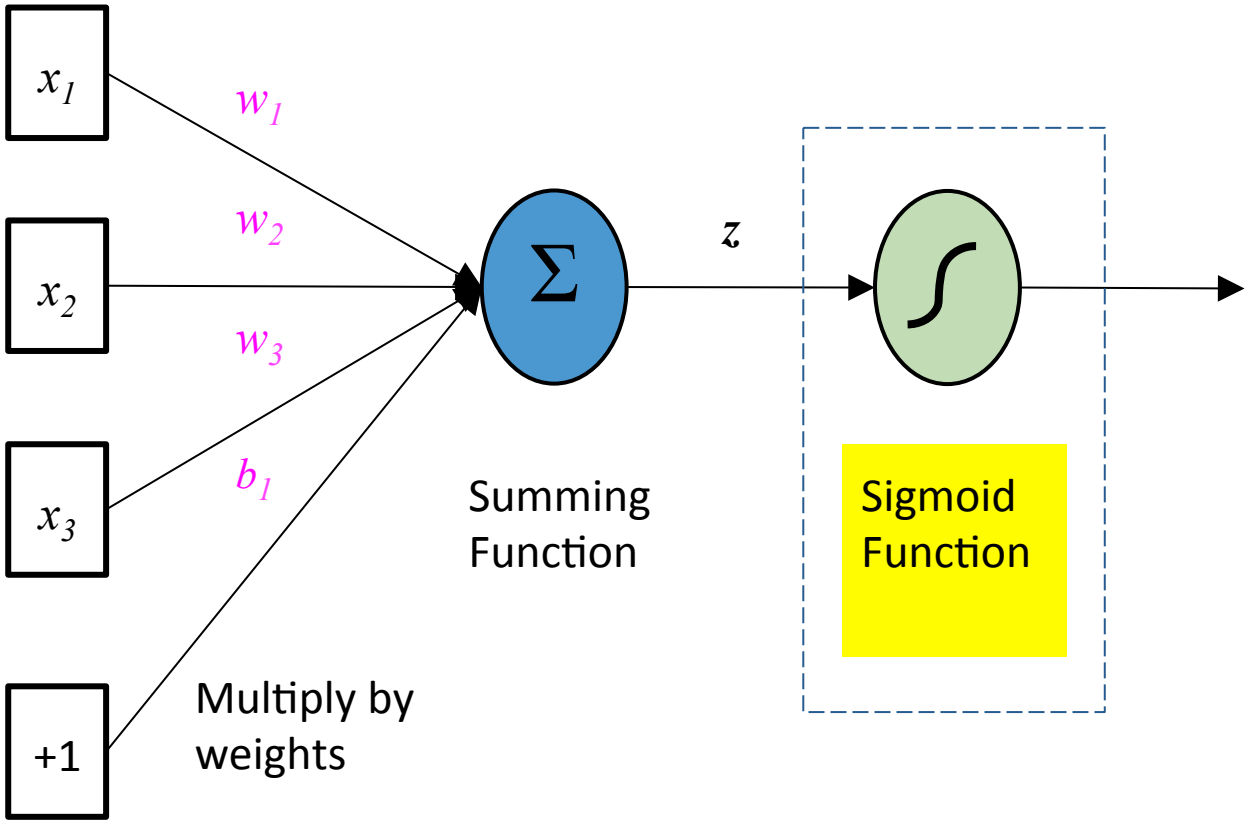
- **Backpropagation!**
  - Repeated application of chain rule of calculus
  - Locally minimize the objective
  - Requires all "blocks" of the network to be differentiable



LeCun et. al. *Efficient Backpropagation*. 1998

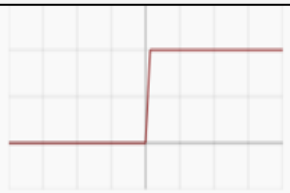
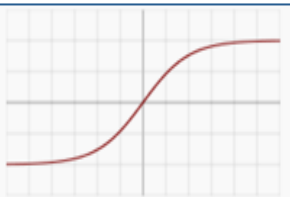
# Nonlinearity Functions

(i.e. transfer or activation functions)



# Nonlinearity Functions

(aka transfer or activation functions)

| Name | Plot | Equation | Derivative ( w.r.t $x$ ) |
|---|---|---|---|
| Binary step |  | $f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for} \quad x \neq 0 \\ ? & \text{for} \quad x = 0 \end{cases}$ |
| Logistic (a.k.a Soft step) |  | $f(x) = \frac{1}{1+e^{-x}}$ | $f'(x) = f(x)(1 - f(x))$ |
| TanH |  | $f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$ | $f'(x) = 1 - f(x)^2$ |
| Rectifier (ReLU)[9] |  | $f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$ |

usually works best in practice

https://en.wikipedia.org/wiki/Activation_function#Comparison_of_activation_functions

# Today

- Deep Learning
  - Basics
  - History
  - Why is this breakthrough ?
  - Recent trends

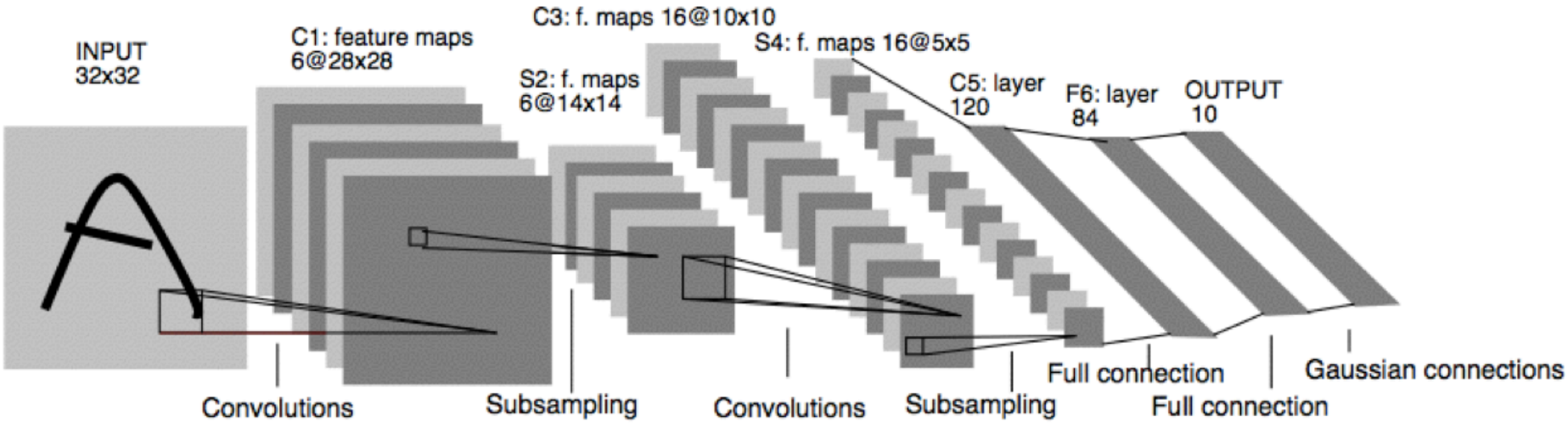
# Many classification models invented since late 80's

- Neural networks
- Boosting
- Support Vector Machine
- Maximum Entropy
- Random Forest
- ……

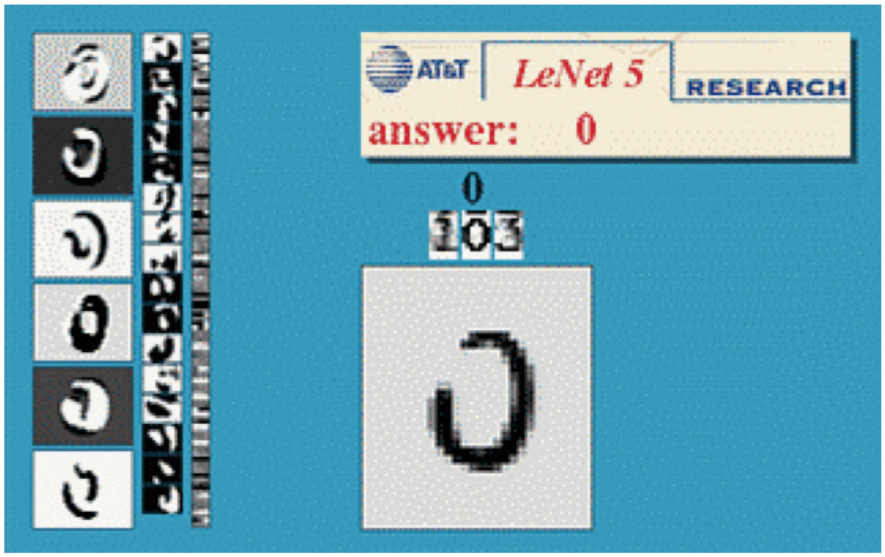
# Deep Learning in the 90's

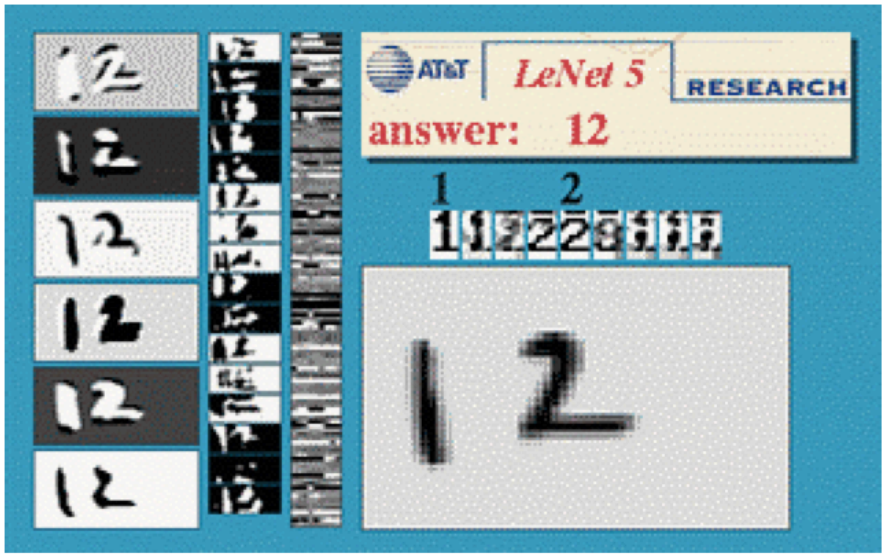
- Prof. Yann LeCun invented Convolutional Neural Networks

- First NN successfully trained with many layers



# “LeNet” Early success at OCR




Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11): 2278–2324, 1998.

# Between ~2000 to ~2011 Machine Learning Field Interest

- Learning with Structures ! + Convex Formulation!
  - Kernel learning
  - Transfer Learning
  - Semi-supervised
  - Manifold Learning
  - Sparse Learning
  - Structured input-output learning …
  - Graphical model

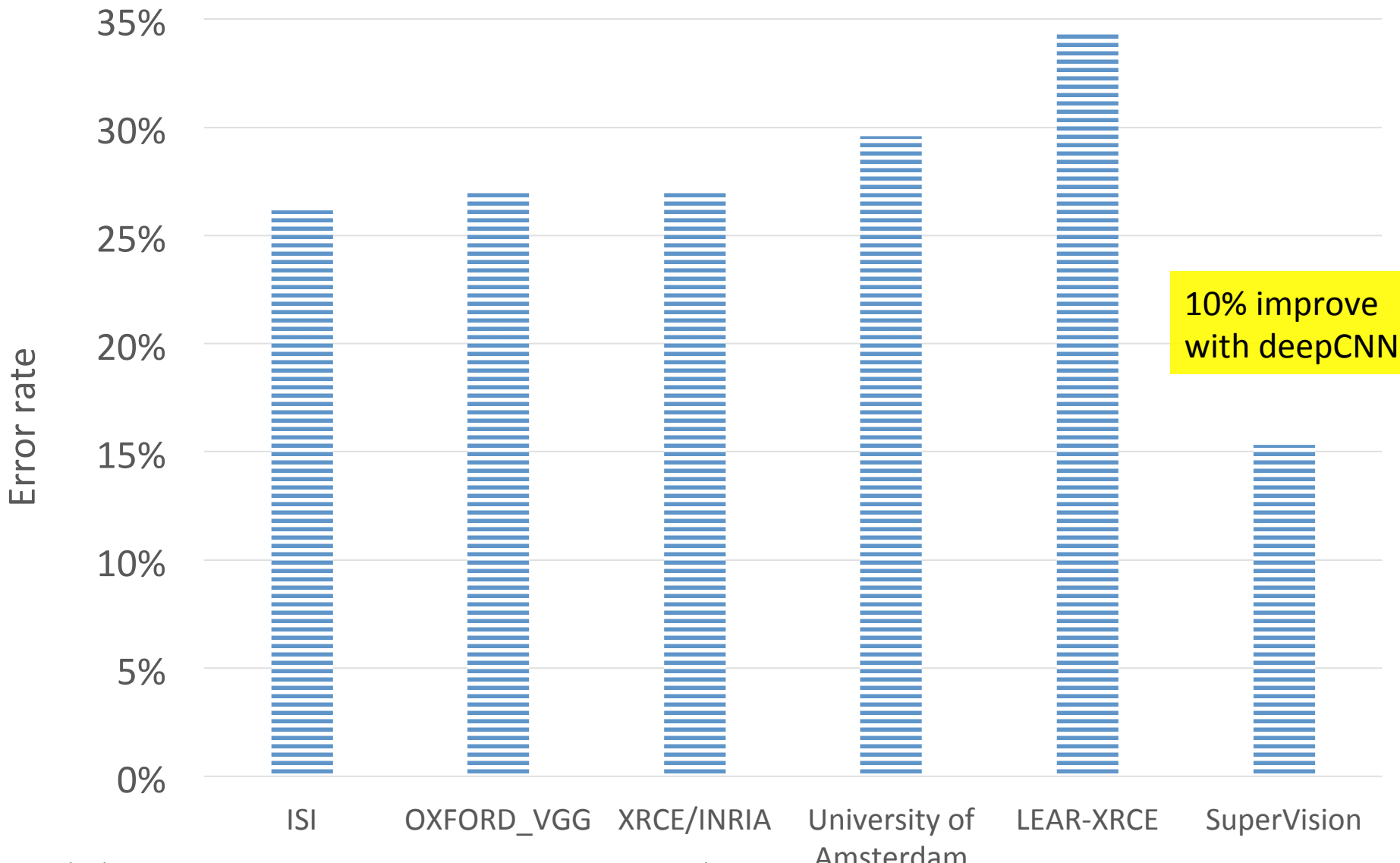
# "Winter of Neural Networks" Since 90's ! to ~2010

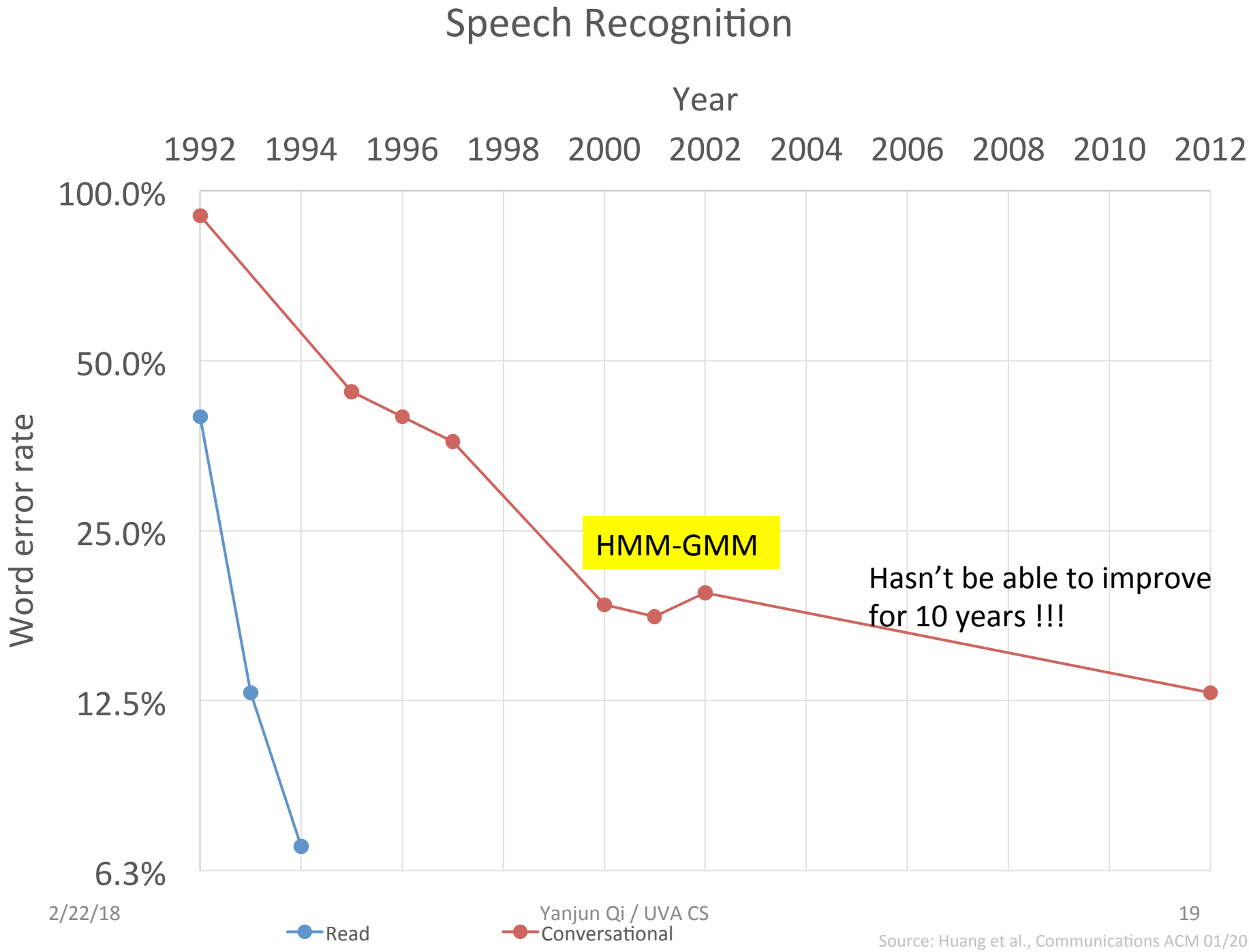
- Non-convex

- Need a lot of tricks to play with
  - How many layers ?
  - How many hidden units per layer ?
  - What topology among layers ? …….

- Hard to perform theoretical analysis

# Large-Scale Visual Recognition Challenge 2012








Source: Huang et al., Communications ACM 01/2014

# Today

- Deep Learning
  - Basics
  - History
  - Why is this breakthrough ?
  - Recent trends

MIT Technology Review

**10 Breakthrough Technologies 2013**

Think of the most frustrating, intractable, or simply annoying problems you can imagine. Now think about what technology is doing to fix them. That's what we did in coming up with our annual list of 10 Breakthrough Technologies. We're looking for technologies that we believe will expand the scope of human possibilities.

Deep Learning

**10 Breakthrough Technologies 2017**

These 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.

Deep Reinforcement Learning


Generative Adversarial Network (GAN)

# WHY BREAKTHROUGH ?

# DNNs help us build more intelligent computers

- Perceive the world,
  - e.g., objective recognition, speech recognition, …
- Understand the world,
  - e.g., machine translation, text semantic understanding
- Interact with the world,
  - e.g., AlphaGo, AlphaZero, self-driving cars, …
- Being able to think / reason,
  - e.g., learn to code programs, learn to search deepNN, …
- Being able to imagine / to make analogy,
  - e.g., learn to draw with styles, ……

# How can we build more intelligent computer / machine ?

- Able to
  - **Perceive the world**
  - **Understand the world**
  - **Interact with the world**

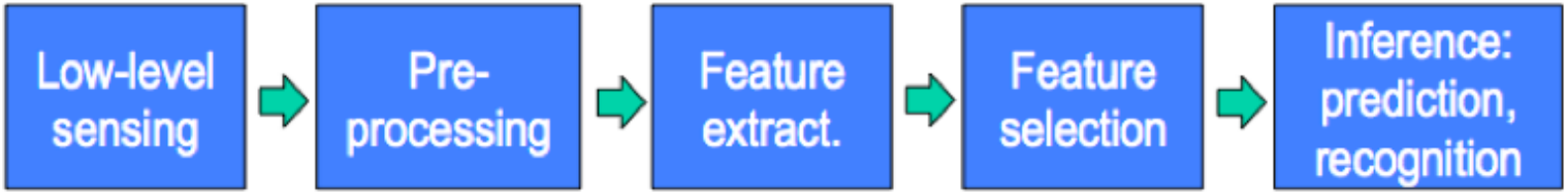
- This needs
  - Basic speech capabilities
  - Basic vision capabilities
  - Language understanding
  - User behavior / emotion understanding
  - Being Able to think / Reason

# Plenty of Data

- Text: trillions of words of English + other languages
- Visual: billions of images and videos
- Audio: thousands of hours of speech per day
- User activity: queries, user page clicks, map requests, etc,
- Knowledge graph: billions of labeled relational triplets
- ……

Dr. Jeff Dean's talk

# Deep Learning Way:
# Learning features / Representation from data




## Feature Engineering

- ✓ Most critical for accuracy
- ✓ Account for most of the computation for testing
- ✓ Most time-consuming in development cycle
- ✓ Often hand-craft and task dependent in practice

## Feature Learning

- ✓ Easily adaptable to new similar tasks
- ✓ Layerwise representation
- ✓ Layer-by-layer unsupervised training
- ✓ Layer-by-layer supervised training

# To perceive the world: Application I: Objective Recognition / Image Labeling


72%, 2010

74%, 2011

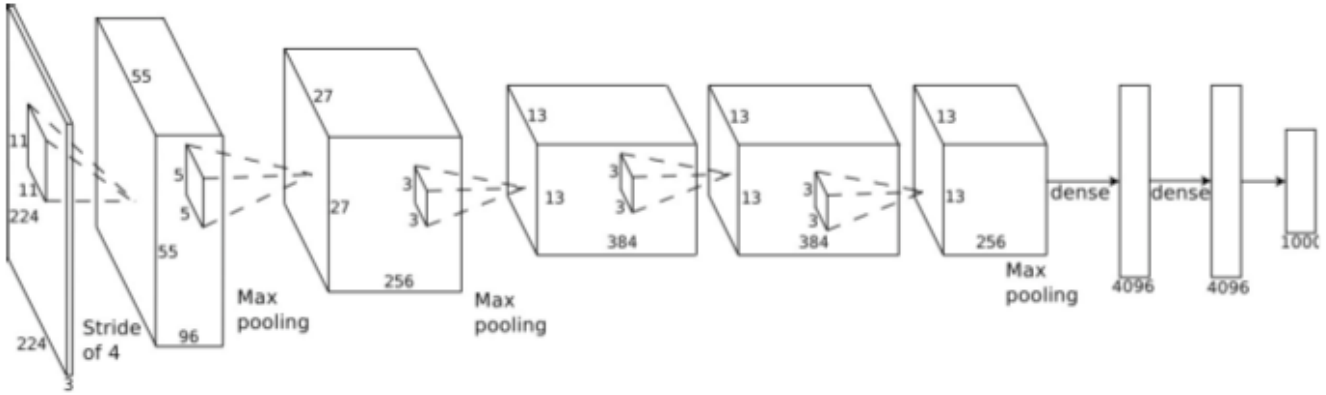
85%, 2012

"Very large-scale" ImageNet competition

(**training on 1.2 million images [X]**

**vs.1000 different word labels [Y]**)

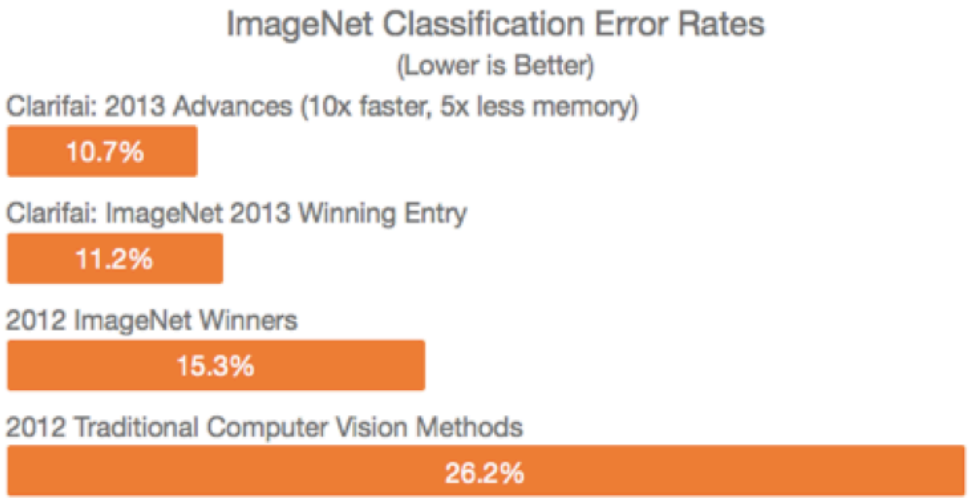
# To perceive the world: Application I: Objective Recognition / Image Labeling




Deep Convolution Neural Network (CNN) and variants have won (as best systems) on "very large-scale" ImageNet competition 2012-2017

# ImageNet Challenge 2013

- Clarifai ConvNet model wins at 11% error rate




- Many other participants used ConvNets
- OverFeat by Pierre Sermanet from NYU: shipped binary program to execute pre-trained models



Olivier Grisel's talk

# ImageNet Challenge 2014

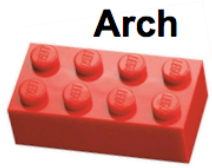
- In the mean time Pierre Sermanet had joined other people from Google Brain
- Monster model: GoogLeNet now at **6.7% error rate**



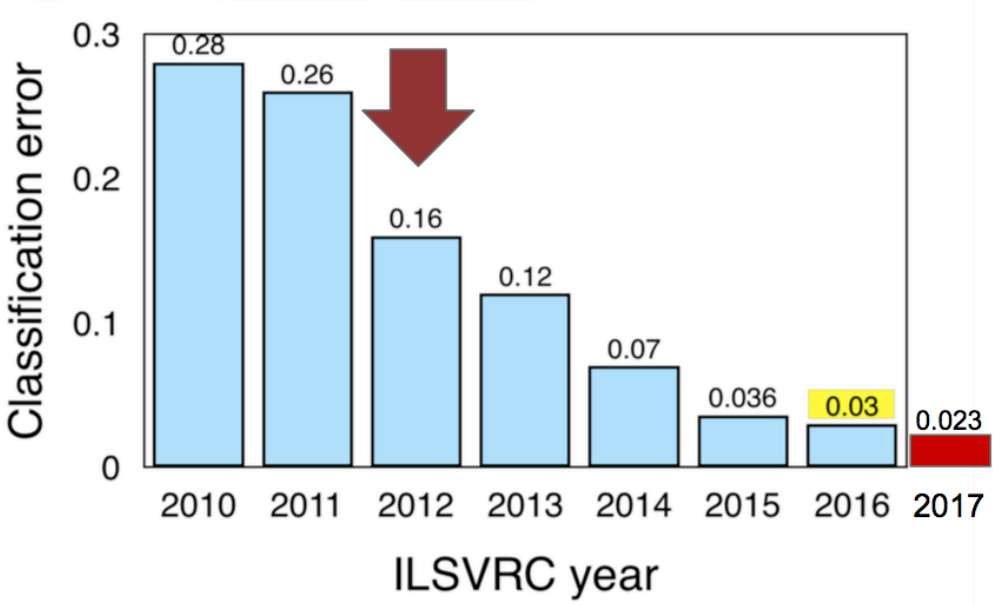
Dr. Jeff Dean's talk

Olivier Grisel's talk

# ImageNet Challenge



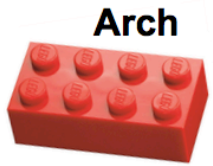

- 2010-11: hand-crafted computer vision pipelines
- 2012-2016: ConvNets
  - 2012: AlexNet
    - major deep learning success
  - 2013: ZFNet
    - improvements over AlexNet
  - 2014
    - VGGNet: deeper, simpler
    - InceptionNet: deeper, faster
  - 2015
    - ResNet: even deeper
  - 2016
    - ensembled networks
  - 2017
    - Squeeze and Excitation Network



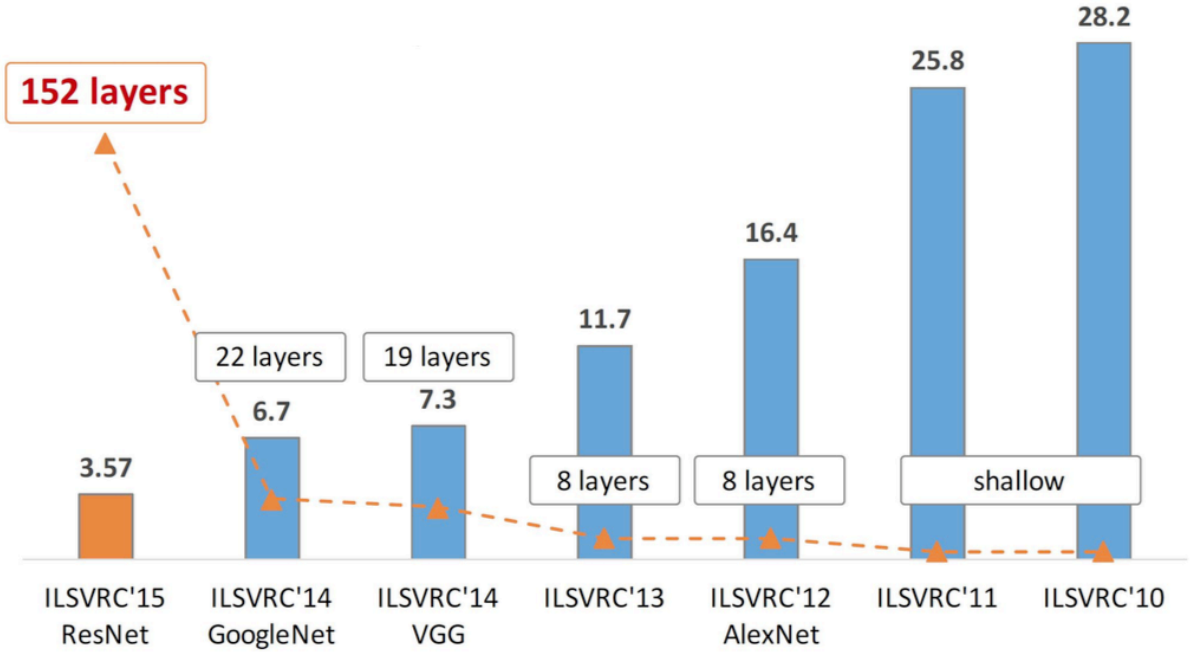



Adapt from From NIPS 2017 DL Trend Tutorial

# Revolution of Depth








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



Adapt from From NIPS 2017 DL Trend Tutorial

# To understand the world: Application II: Semantic Understanding

## e.g. Word embedding / Neural Language Models

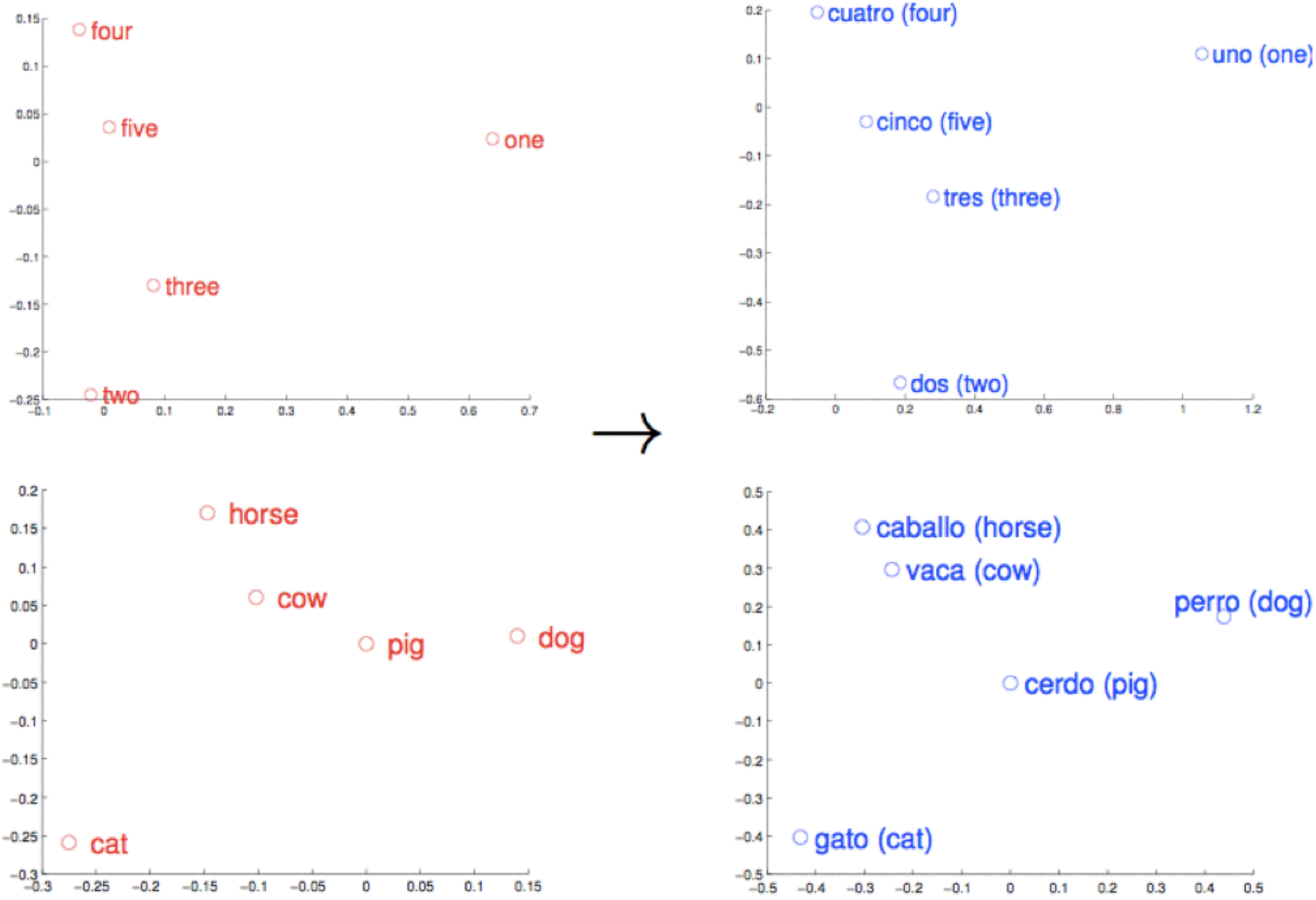
- Learn to embed each word into a vector of real values
  - Semantically similar words have closer embedding representations

- Progress in 2013/14
  - Can uncover semantic /syntactic word relationships
    - [king] - [male] + [female] ~= [queen]
    - [Berlin] - [Germany] + [France] ~= [Paris]
    - [eating] - [eat] + [fly] ~= [flying]



Dr. Li Deng's talk

Dr. Jeff Dean's talk




source: Exploiting Similarities among Languages for MT

# To interact with the world: Application III: Deep Learning to Play Games

- DeepMind:
  - Learning to Play & win dozens of Atari games
  - new Deep Reinforcement Learning algorithms
  - AlphaGo / AlphaZero



Olivier Grisel's talk

# Able to Reason: Application IV:

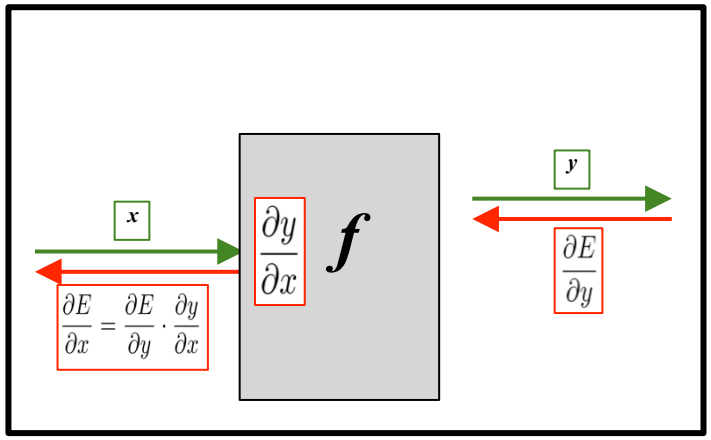
## Deep Learning to Execute and Program

# Neural Turing Machines

- Google DeepMind, October 2014 (very new)
- Neural Network coupled to external memory (tape)
- Analogue to a Turing Machine but differentiable
- Can be used to learn to simple programs from example input / output pairs

Olivier Grisel's talk

# Building Deep Neural Nets

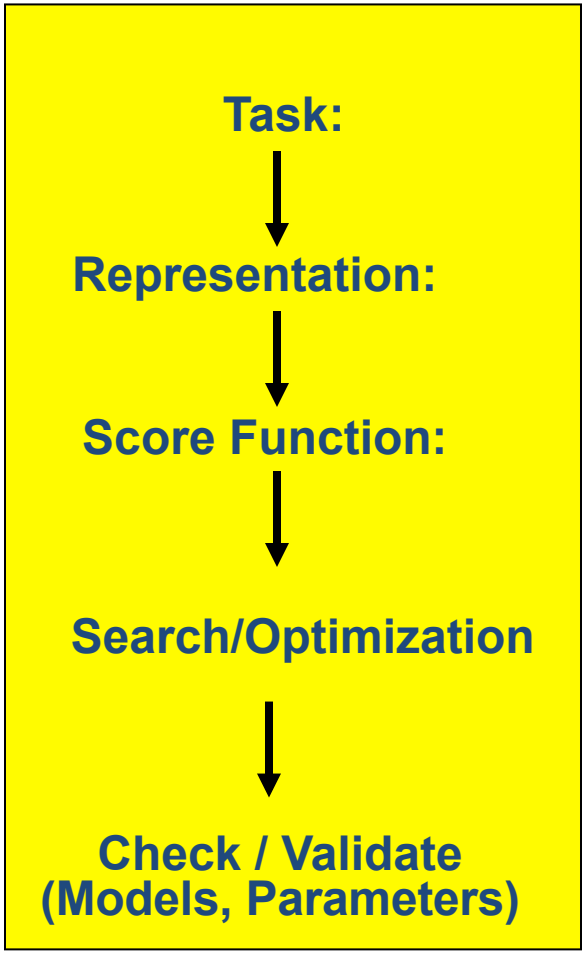



http://cs231n.stanford.edu/slides/winter1516_lecture5.pdf

# Today

- Deep Learning
  - Basics
  - History
  - Why is this breakthrough ?
  - Recent trends

# Machine (Deep) Learning in a Nutshell



# DESIGN Deep NN: Five Aspects

- Tasks:
  - Generative / Discriminative / Reinforce / Reasoning …
- Formulate Input / Output:
  - Data representation
- Architecture Design:
  - Network Topology, Network Parameters
- Training / Searching / Learning
  - With new losses / with new optimization tricks
  - New formulation of learning
  - Scaling up with GPU, Scaling up with distributed optimization , e.g. Asynchronous SGD
- Validation / Trust / Test / Understand …

# Some Trends after ~2010

More Details and Materials from
My 2017 Fall Advanced Deep Learning
https://qdata.github.io/deep2Read/

Inputs and Outputs

Losses

Architectures:

Validation



Adapt from From NIPS 2017 DL Trend Tutorial

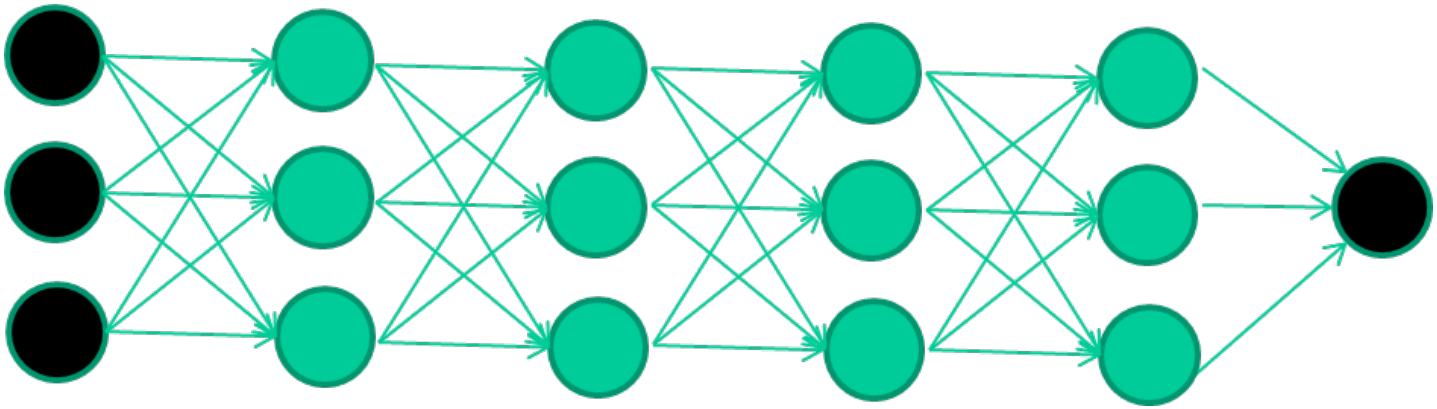
# Some Recent Trends

- 1. Autoencoder / layer-wise training
- 2. CNN / Residual / Dynamic parameter
- 3. RNN / Attention / Seq2Seq, …
- 4. Neural Architecture with explicit memory
- 5. NTM 4program induction / sequential decisions
- 6. Learning to optimize / Learning DNN architectures
- 7. Learning to learn / meta-learning/ few-shots
- 8. DNN on graphs / trees
- 9. Deep Generative models, e.g., autoregressive
- 10. Generative Adversarial Networks (GAN)
- 11. Deep reinforcement learning
- 12. Validate / Evade / Test / Understand / Verify DNNs

# Recent Trend (1): Layer-wise pretraining / Auto-Encoder

**Losses**

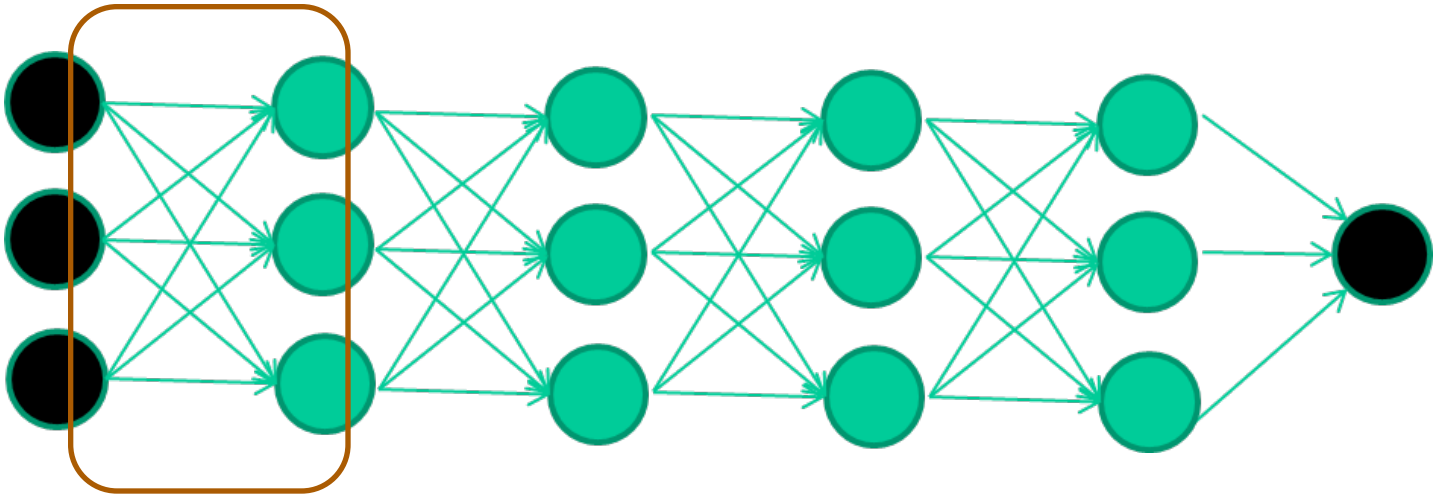
# The new way to train multi-layer NNs…





https://www.macs.hw.ac.uk/~dwcorne/Teaching/introdl.ppt

# The new way to train multi-layer NNs…

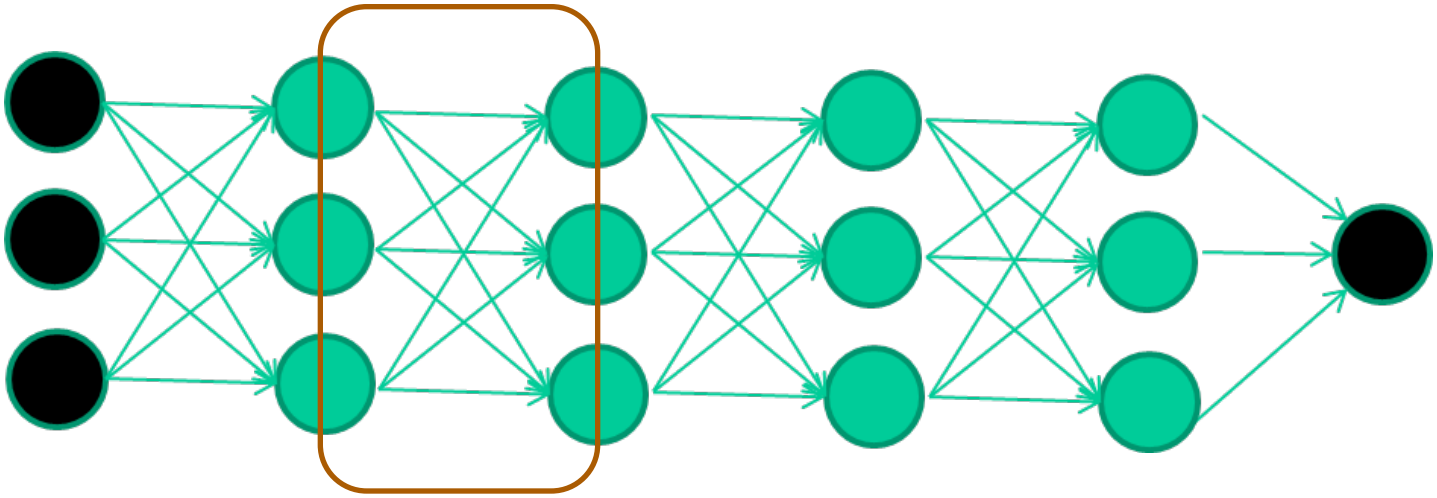


Train **this** layer first



https://www.macs.hw.ac.uk/~dwcorne/Teaching/introdl.ppt

# The new way to train multi-layer NNs…



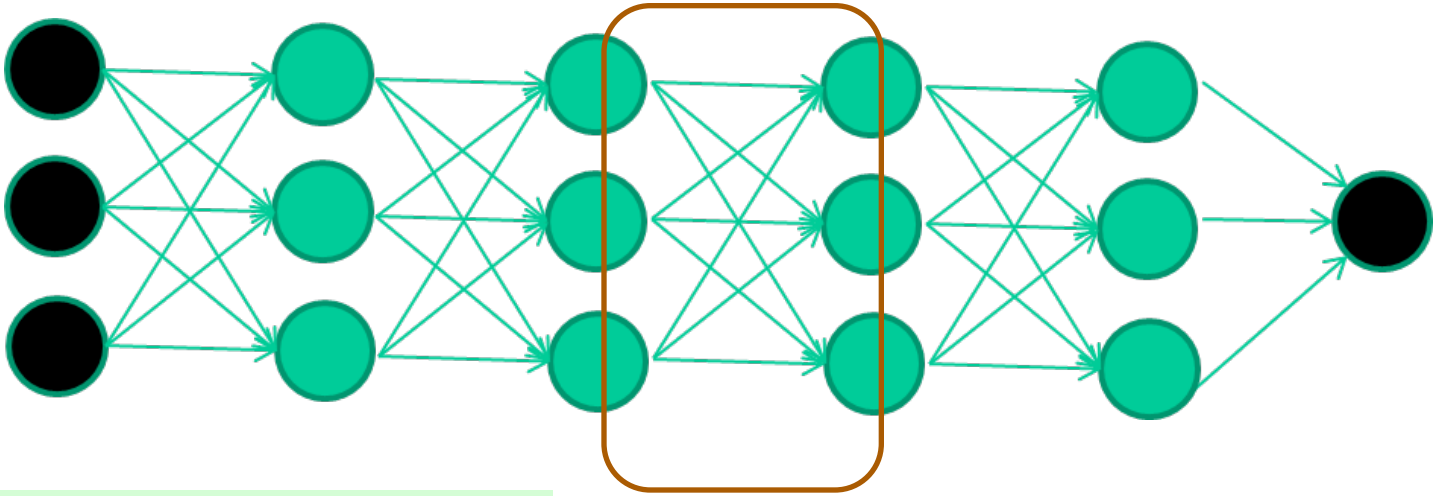
Train **this** layer first

then **this** layer



https://www.macs.hw.ac.uk/~dwcorne/Teaching/introdl.ppt

# The new way to train multi-layer NNs…



Train **this** layer first

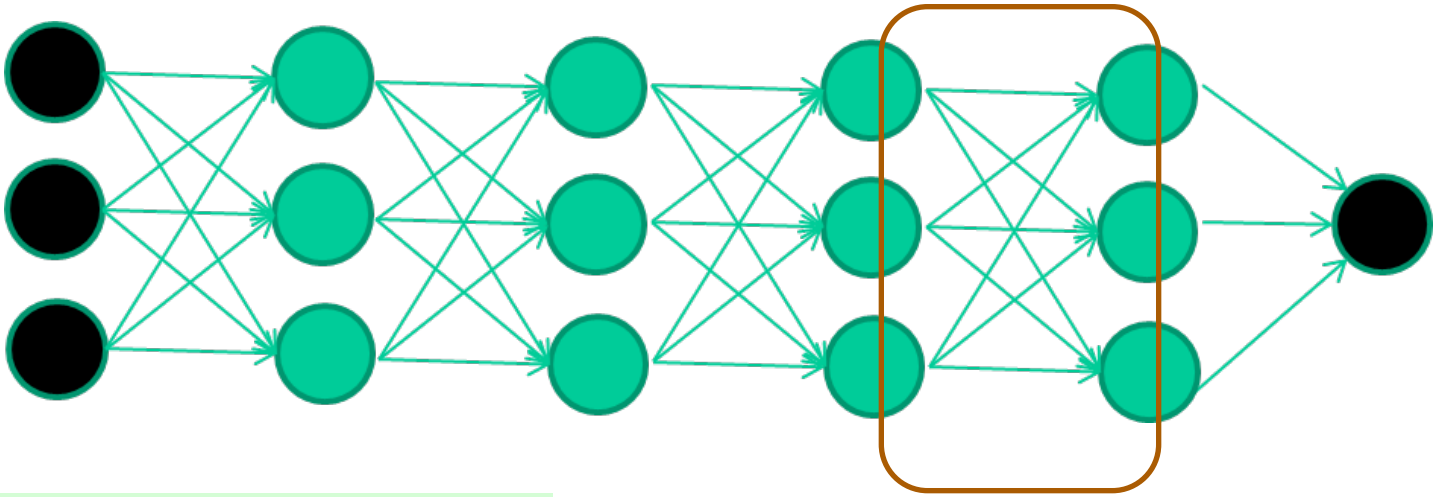
then **this** layer

then **this** layer



https://www.macs.hw.ac.uk/~dwcorne/Teaching/introdl.ppt

# The new way to train multi-layer NNs…



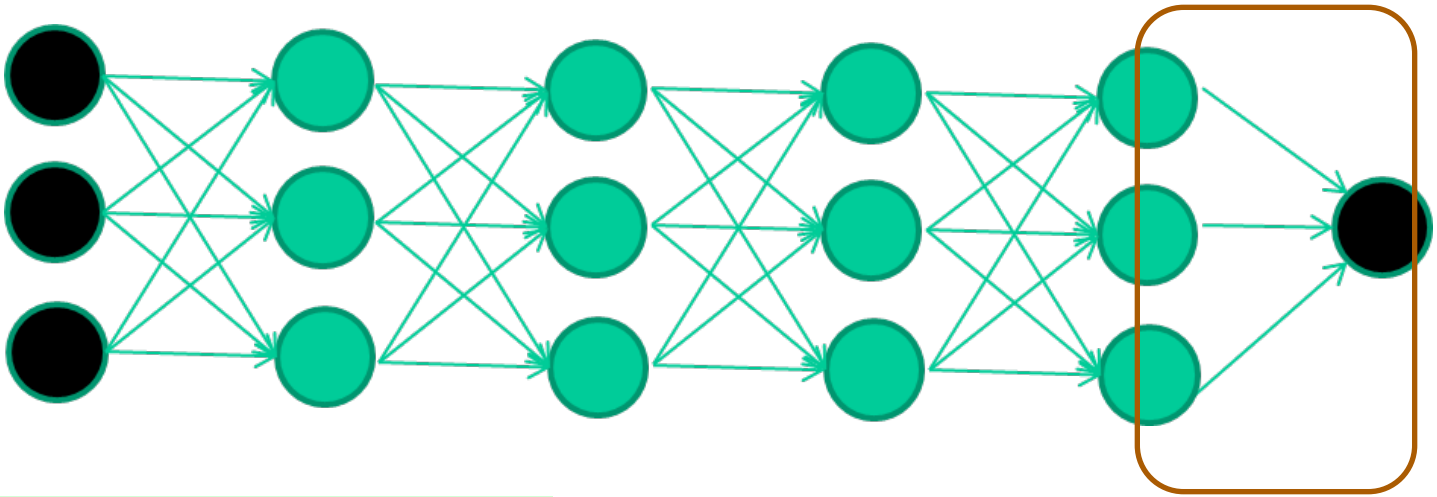
Train **this** layer first

then **this** layer

then **this** laver

then **this** layer

# The new way to train multi-layer NNs…



Train **this** layer first

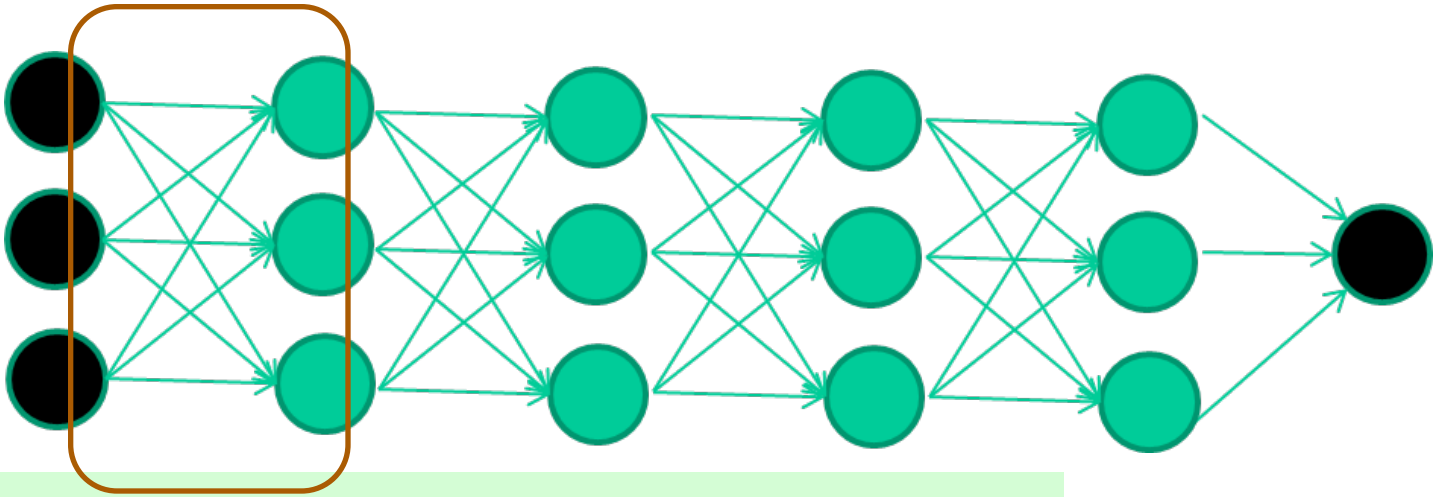
then **this** layer

then **this** layer

then **this** layer

finally **this** layer

# The new way to train multi-layer NNs…

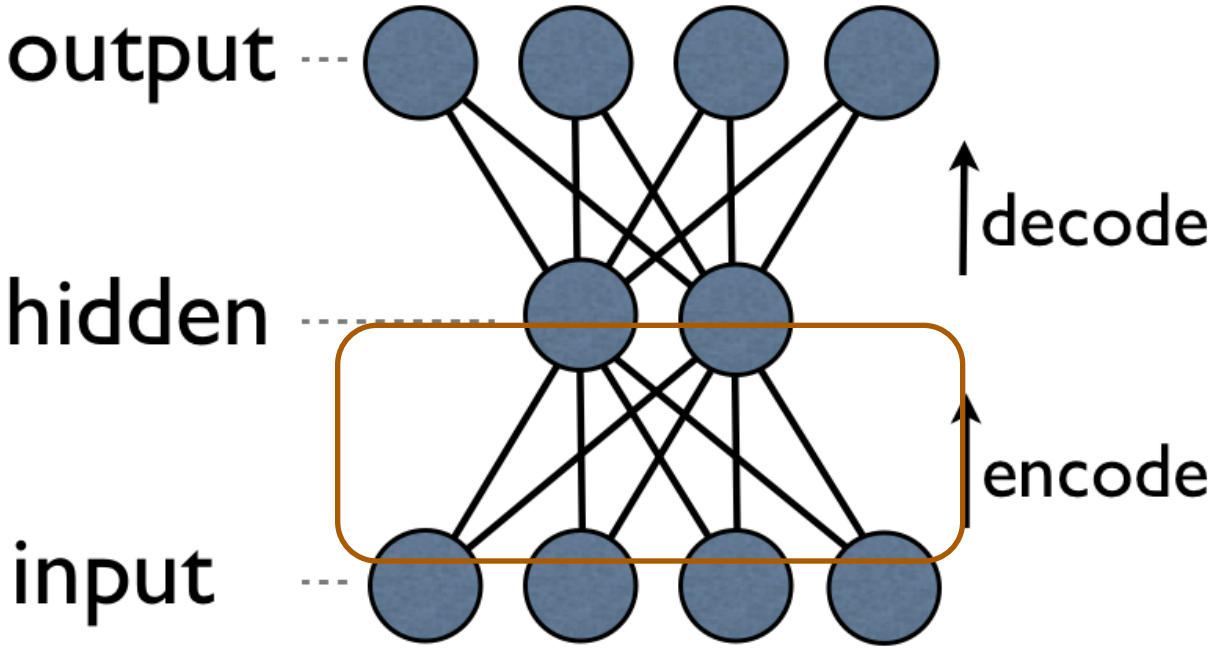


*EACH of the (non-output) layers is trained*

*to be an* ***auto-encoder***

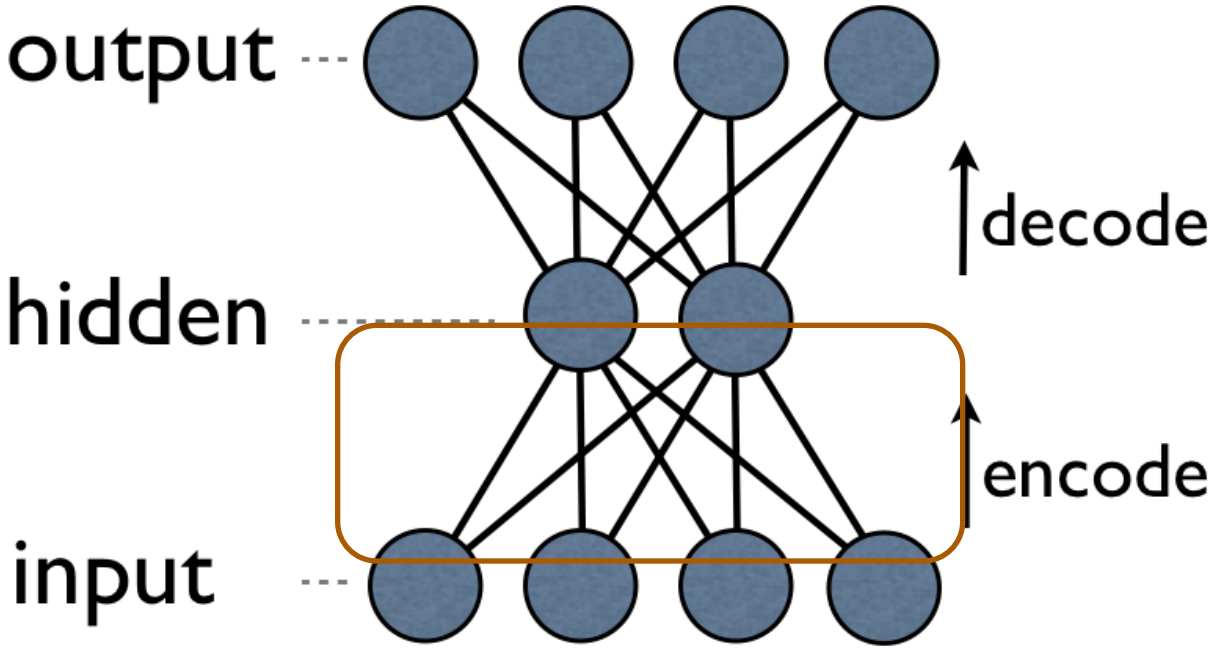
*Basically, it is forced to learn good features that describe what comes from the previous layer*

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



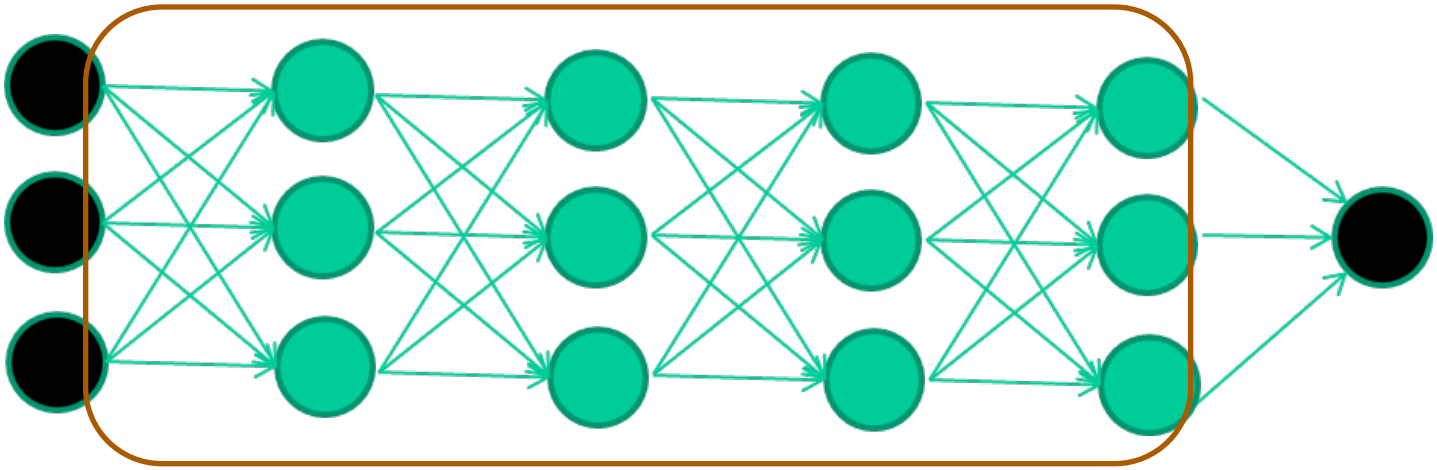

**Reconstruction Loss**

force the ‘hidden layer’ units to become good / reliable feature detectors




**Reconstruction Loss**

# intermediate layers can each be trained to be auto encoders (or similar)



# Recent Trend (2): Convolutional Neural Networks (aka CNNs and ConvNets)

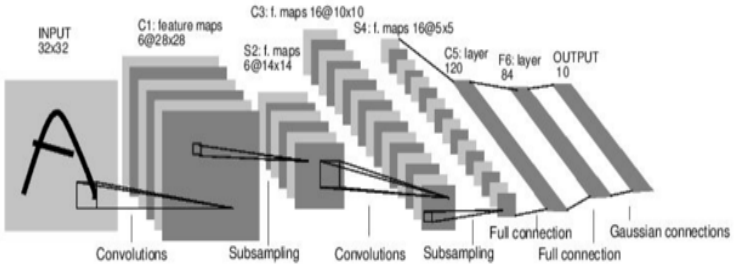
**Architectures:**

- Convolutions

# History of ConvNets

1998

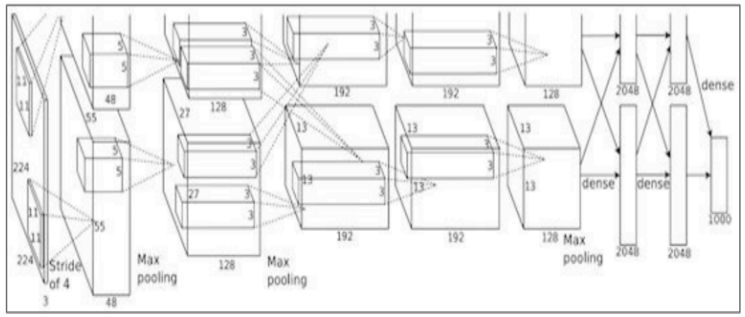
*Gradient-based learning applied to document recognition* [LeCun, Bottou, Bengio, Haffner]




LeNet-5

2012

*ImageNet Classification with Deep Convolutional Neural Networks* [Krizhevsky, Sutskever, Hinton, 2012]




"AlexNet"

# Locality and Translation Invariance



The bird occupies a local area and looks the same in different parts of an image.
**We should construct neural nets which exploit these properties!**



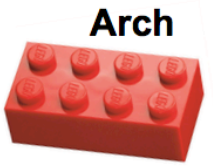
Adapt from From NIPS 2017 DL Trend Tutorial

# Locality and Translation Invariance

- **Locality**: objects tend to have a local spatial support
- **Translation invariance**: object appearance is independent of location

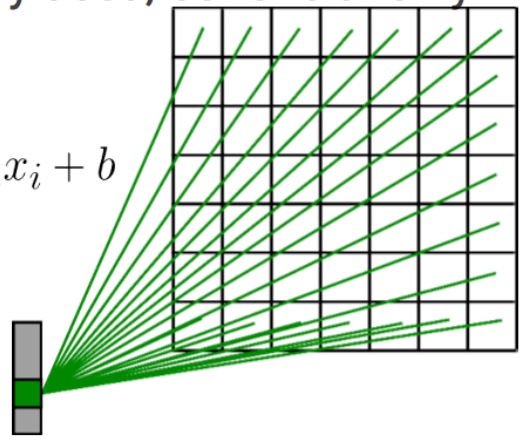
- Can define these properties since an image lies on a grid/lattice
  - ConvNet machinery applicable to other data with such properties, e.g. audio/text

# Incorporating Assumptions: locality



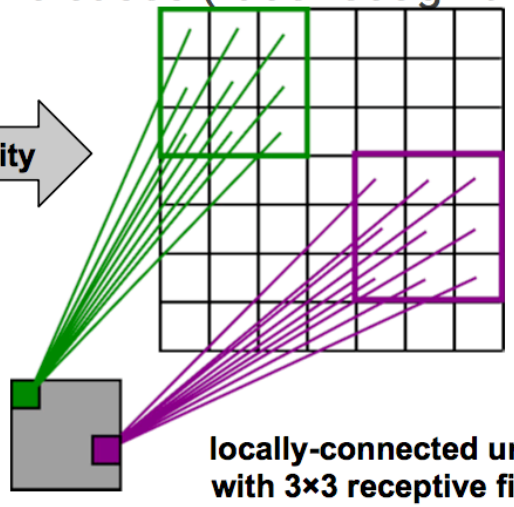

- Make **fully**-connected layer **locally**-connected
- Each unit/neuron is connected to a local rectangular area – receptive field
- Different units connected to different locations
  - output (“**feature map**”) lies on a grid itself
- Rarely used, beneficial only in specific cases (face recognition)

$$y = \sum_{i \in image} w_i x_i + b$$



fully-connected unit

locality



locally-connected units
with 3×3 receptive field

$$y = \sum_{i \in receptive\ field} w_i x_i + b$$

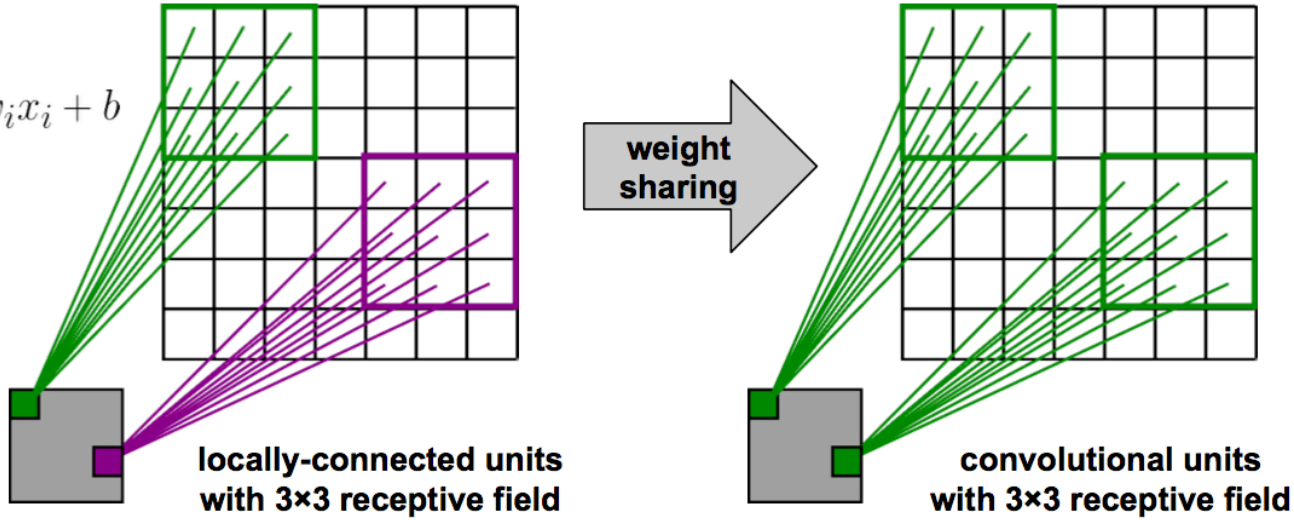


Adapt from From NIPS 2017 DL Trend Tutorial

# Incorporating Assumptions: translation invariance

- **Weight sharing**
  - units connected to different locations have the same weights
  - equivalently, each unit is applied to all locations
- *Convolutional layer – locally-connected layer with weight sharing (translation invariance)*
- The weights are invariant, the output is equivariant

$$y = \sum_{i \in receptive\ field} w_i x_i + b$$



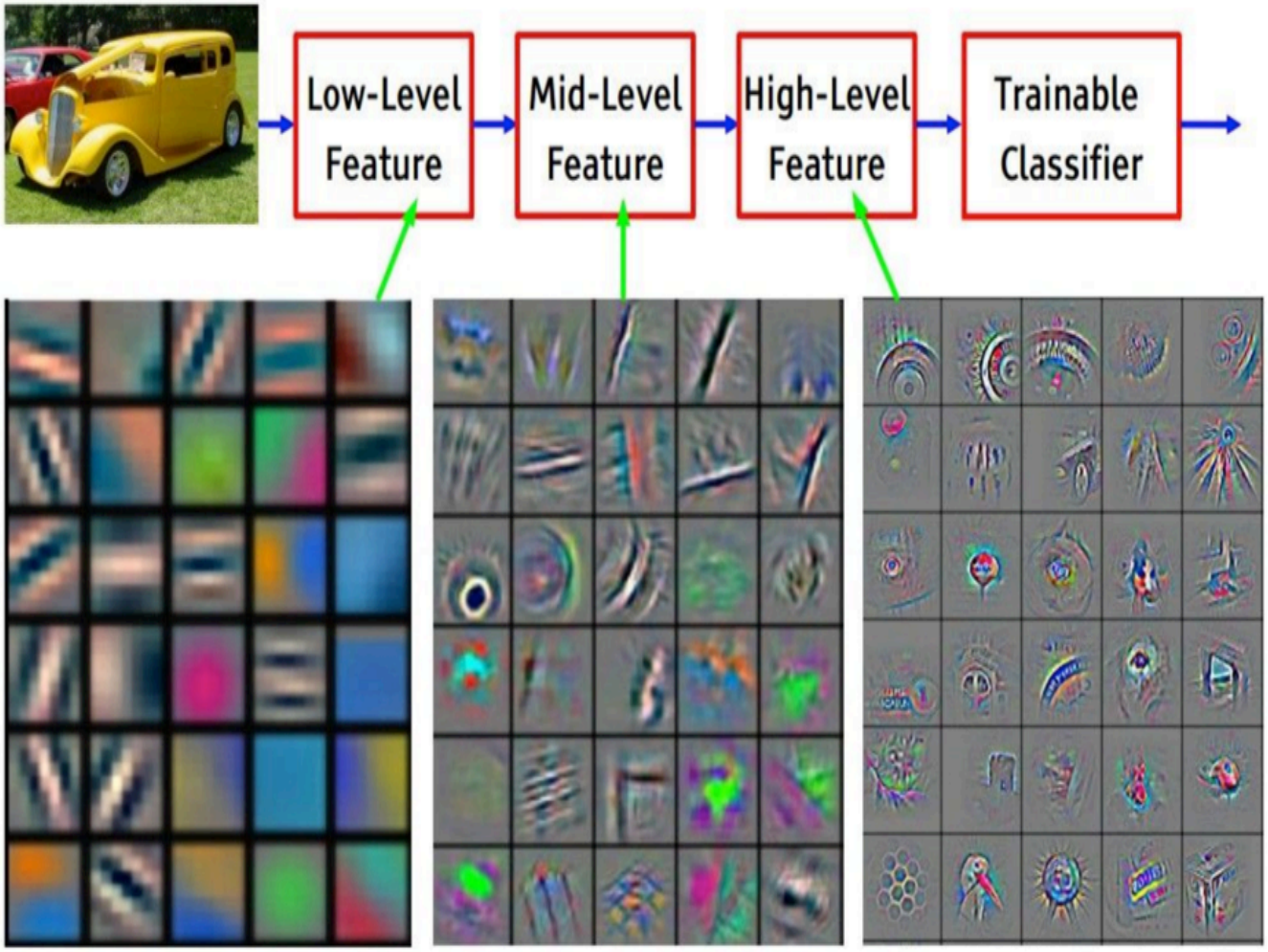

$$y = w * x + b$$



Adapt from From NIPS 2017 DL Trend Tutorial

# Convolutional Neural Networks

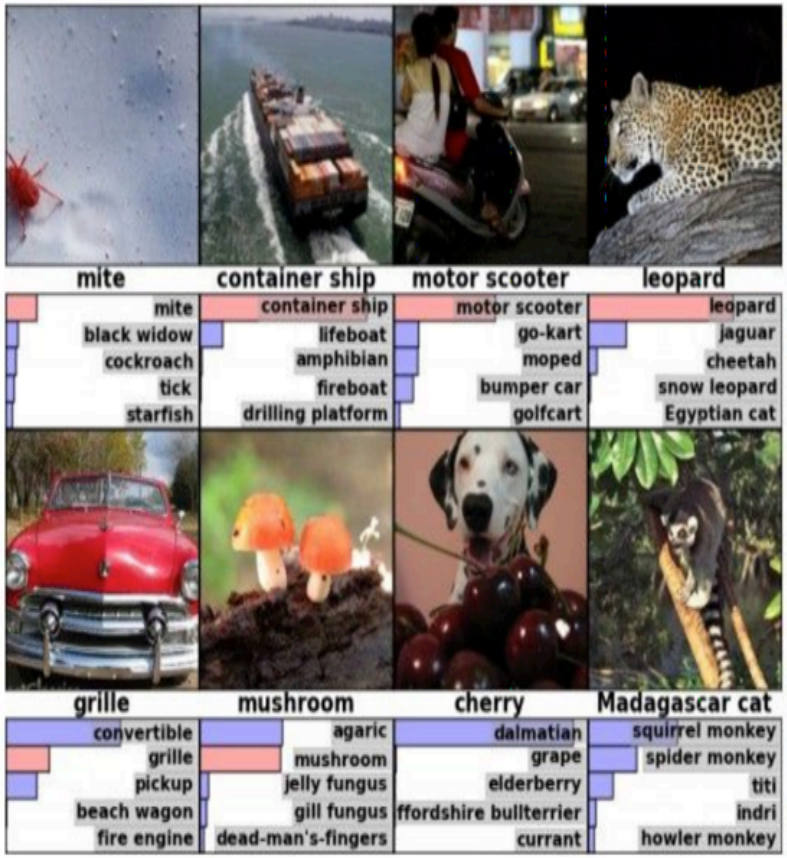
*[From recent Yann LeCun slides]*




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

# Fast-forward to today: ConvNets are everywhere

Classification

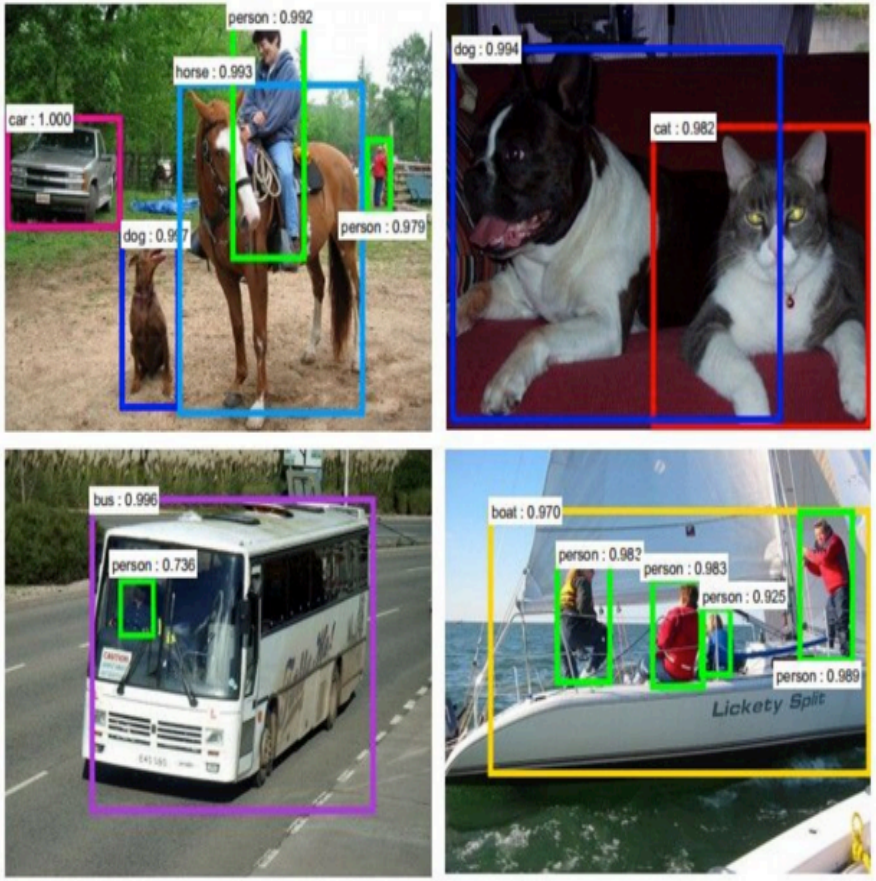



Retrieval

*[Krizhevsky 2012]*

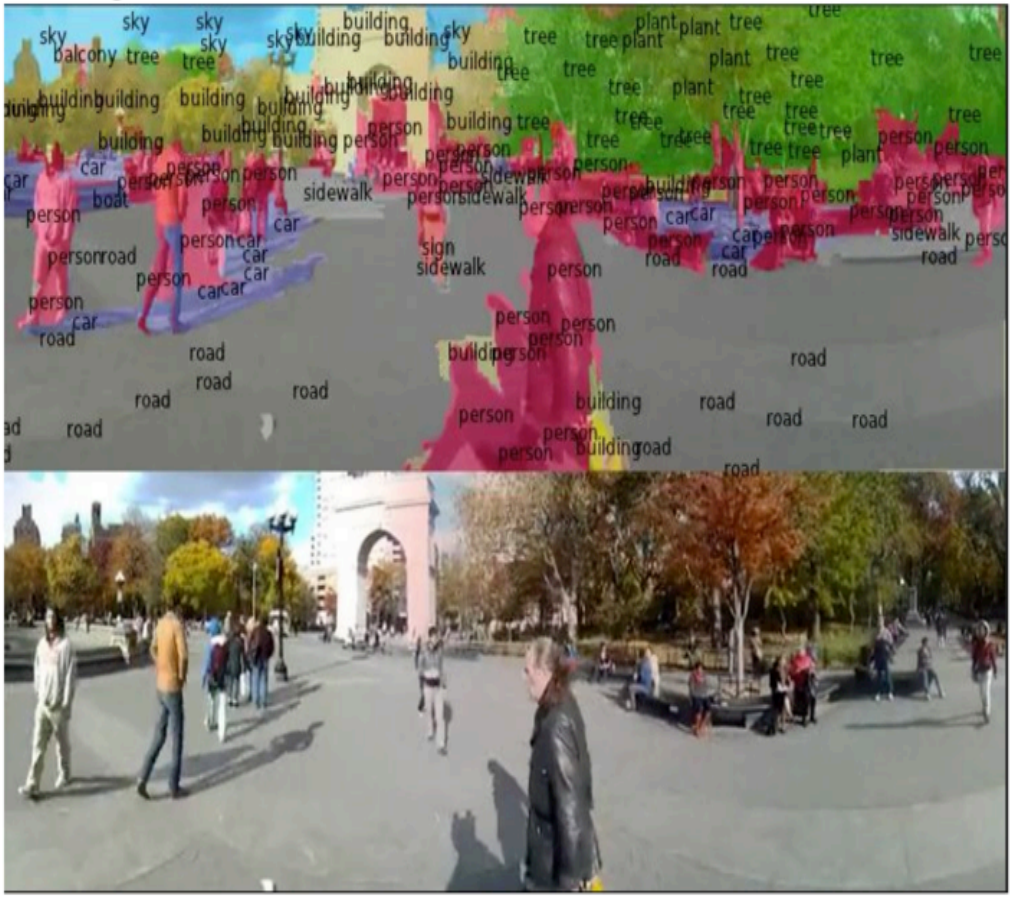
# Fast-forward to today: ConvNets are everywhere

Detection




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

Segmentation




*[Farabet et al., 2012]*

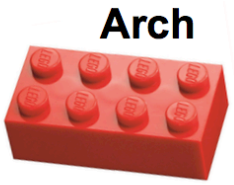
# Fast-forward to today: ConvNets are everywhere


self-driving cars

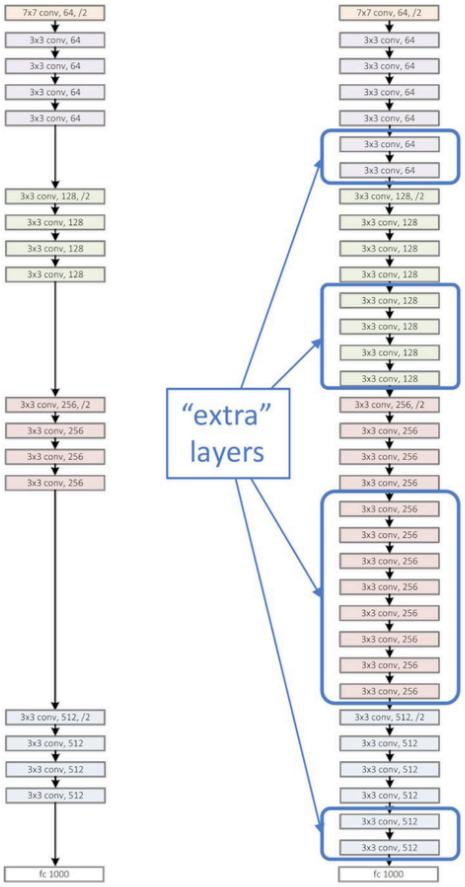

NVIDIA Tegra X1

**Residual Trick:**

# Residual/Skip Connections




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.

# Adaptive / Dynamic Trick:

- Diet Networks:  Thin Parameters for Fat Genomics, ICLR 2017
- Dynamic Filter Networks, NIPS 2016
- Hyper Networks, ICLR 2017
- Optimal Architectures in a Solvable Model of Deep Networks, NIPS16
- AdaNet: Adaptive Structural Learning of Artificial Neural Networks, ICML17
- SplitNet: Learning to Semantically Split Deep Networks for Parameter Reduction and Model Parallelization, ICML17
- Image Question Answering using Convolutional Neural Network with Dynamic Parameter , CVPR 2016
- Many others..

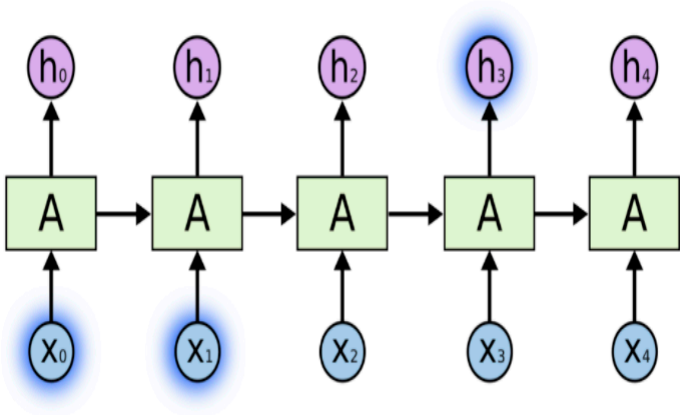
# Recent Trend (3): Recurrent Neural Networks

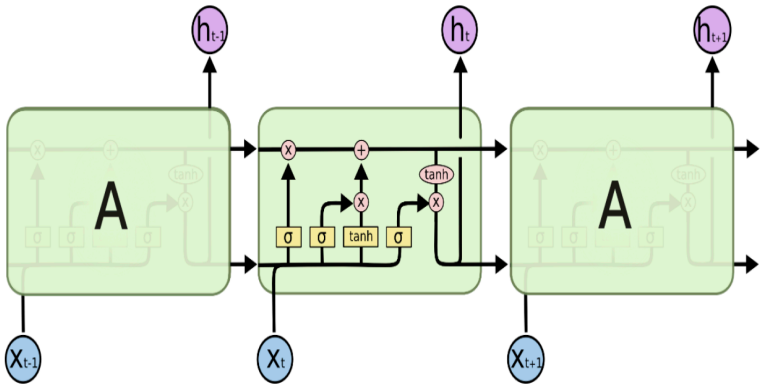
**Architectures:**

- Recurrent, over space and/or time.
  - + attention
- Attention-only!

# Important **Block**: Recurrent Neural Networks (RNN)

- Prof. Schmidhuber invented "Long short-term memory" – Recurrent NN (LSTM-RNN) model in 1997




The repeating module in an LSTM contains four interacting layers.

Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780.



Image Credits from Christoph

# RNN models dynamic temporal dependency

- Make fully-connected layer model each unit recurrently
- Units form a directed chain graph along a sequence
- Each unit uses recent history and current input in modeling

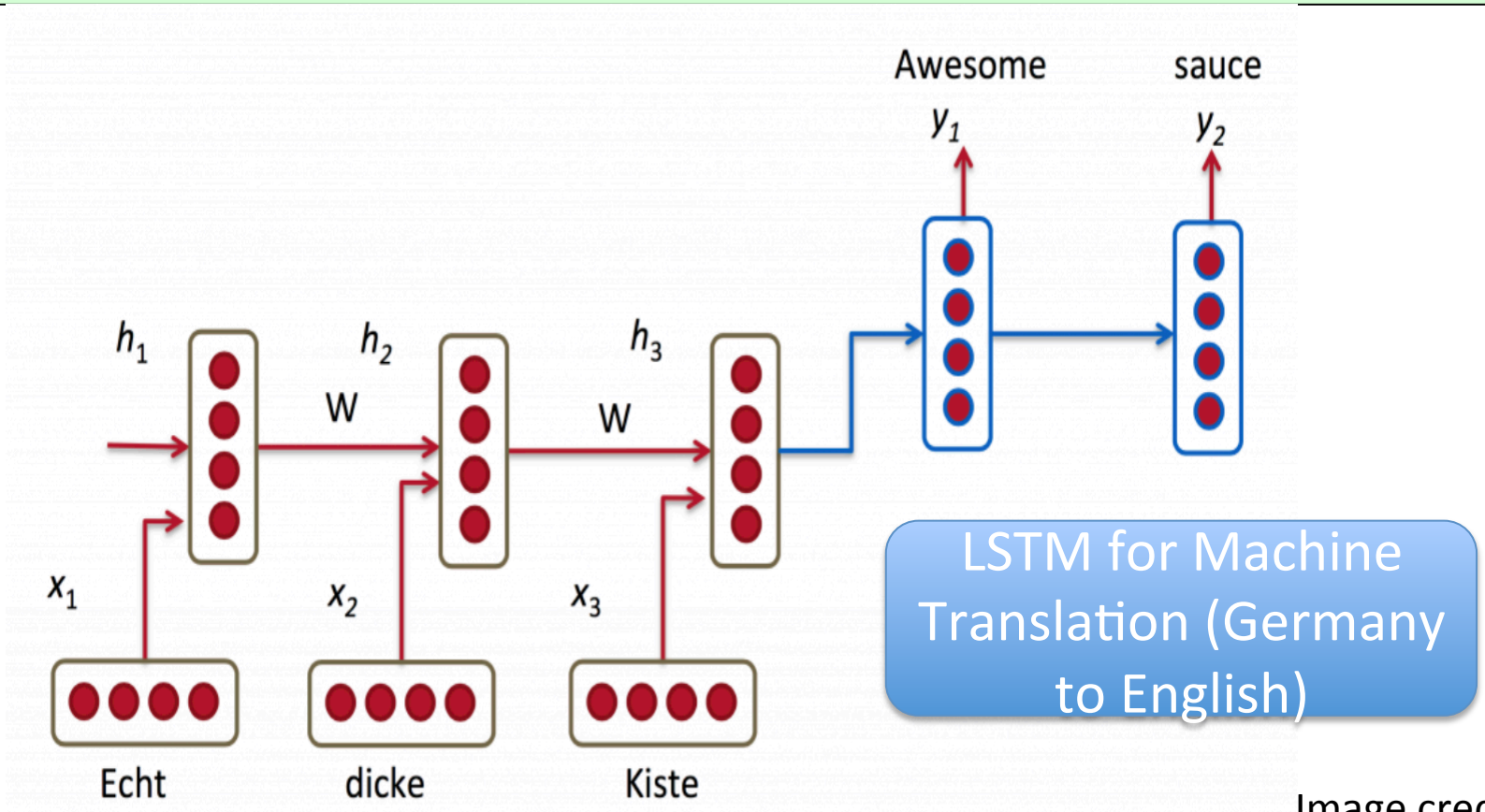
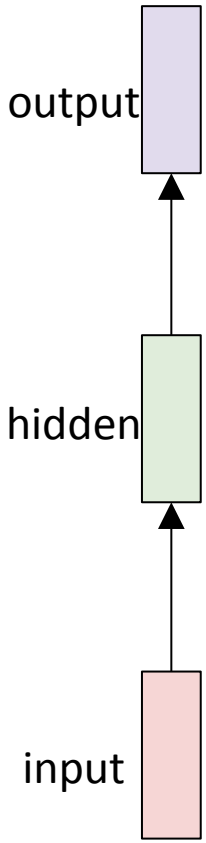




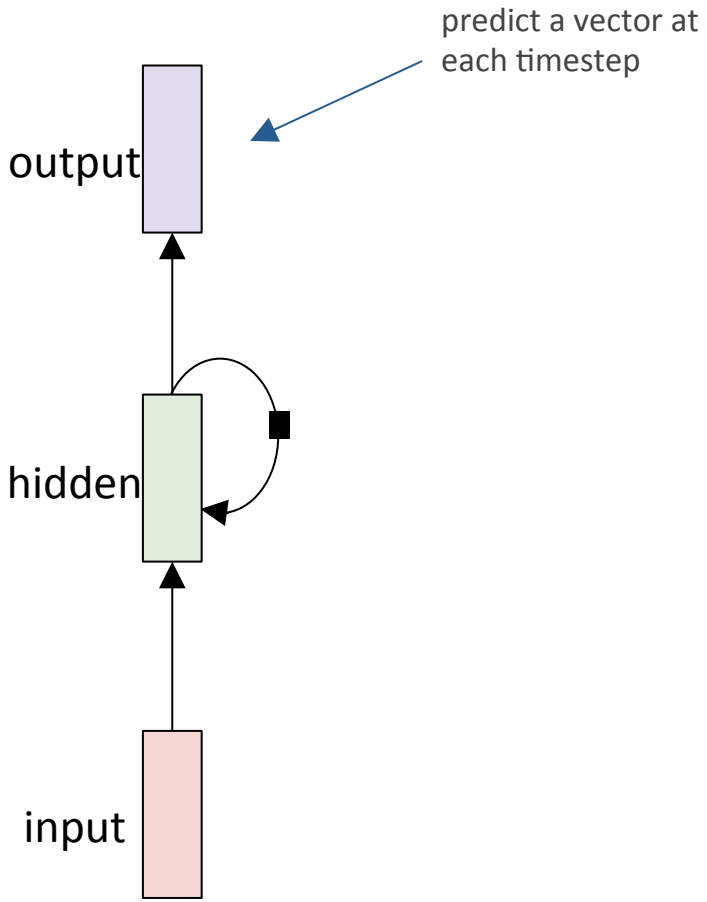

Image credit : wildML

# Recurrent Neural Networks (RNNs)

**Traditional "Feed Forward" Neural Network**



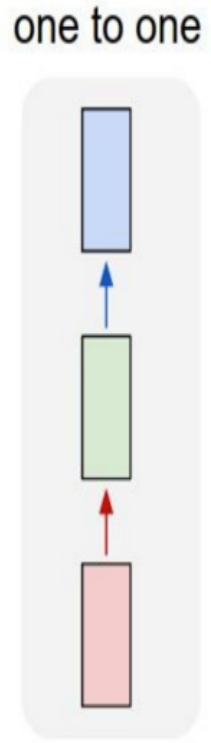

**Recurrent Neural Network**





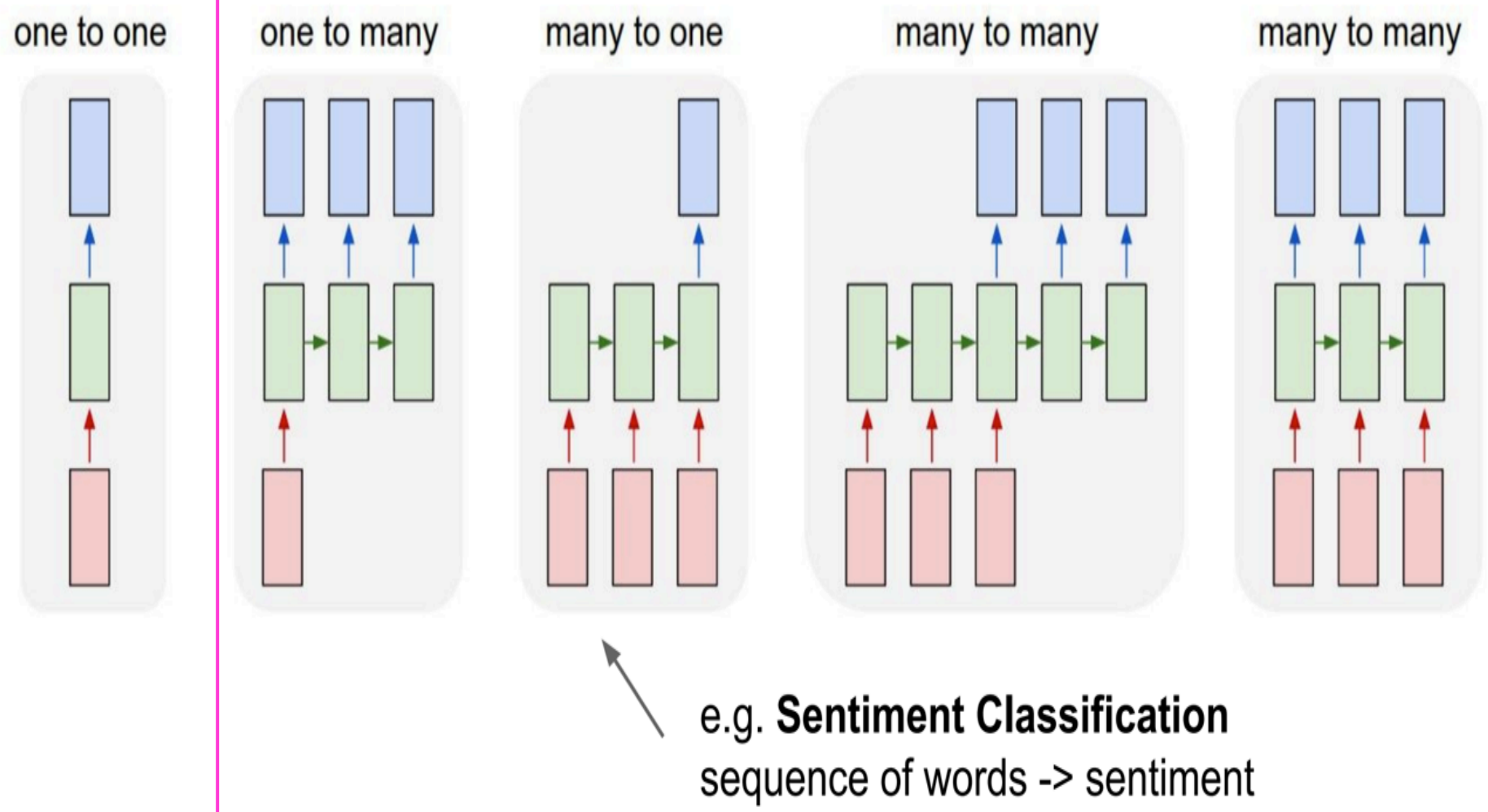

http://cs231n.stanford.edu/slides/

# Standard “Feed-Forward” Neural Network



# Recurrent Neural Networks (RNNs)

RNNs can handle

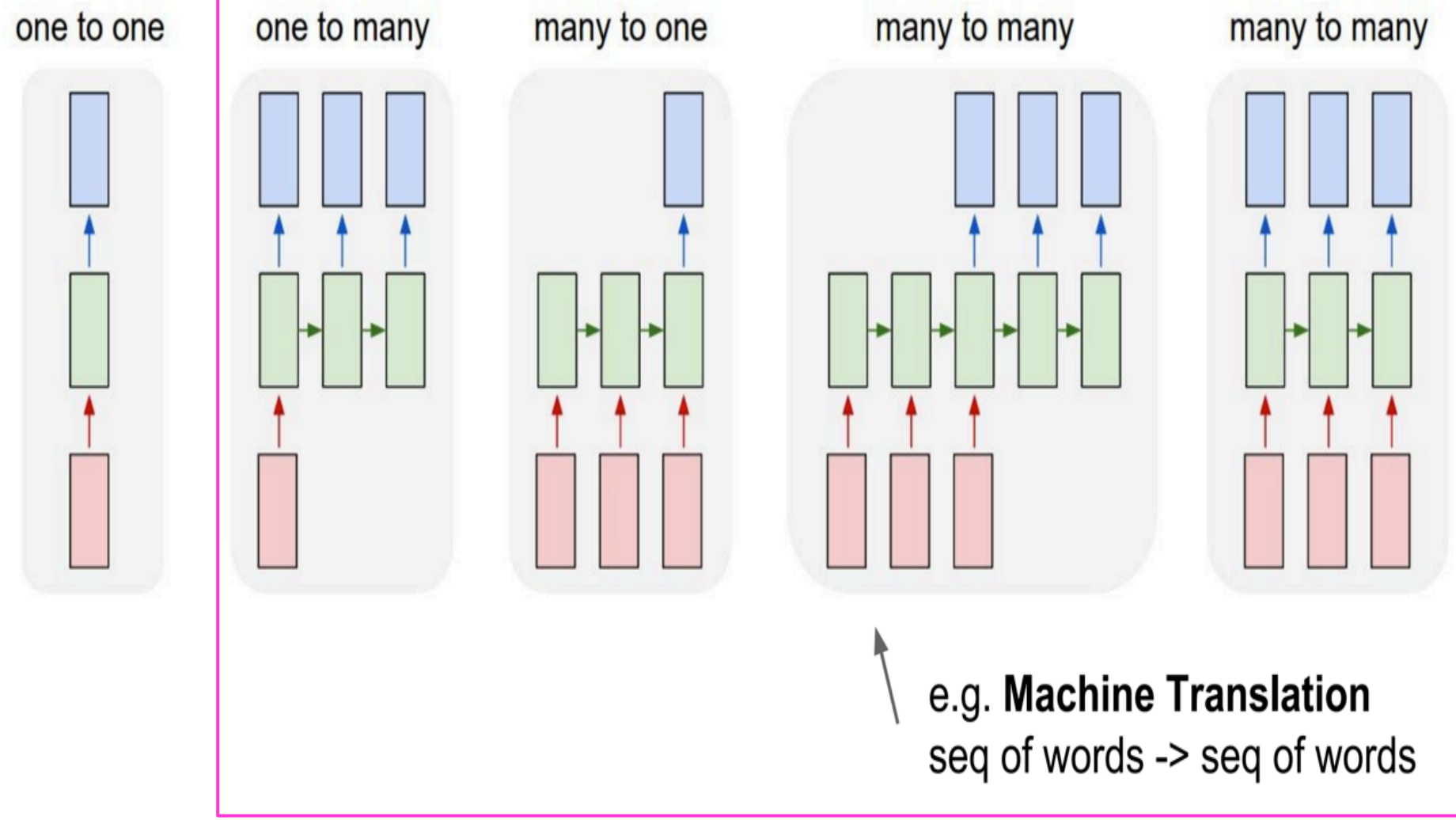





http://cs231n.stanford.edu/slides/

# Recurrent Neural Networks (RNNs)

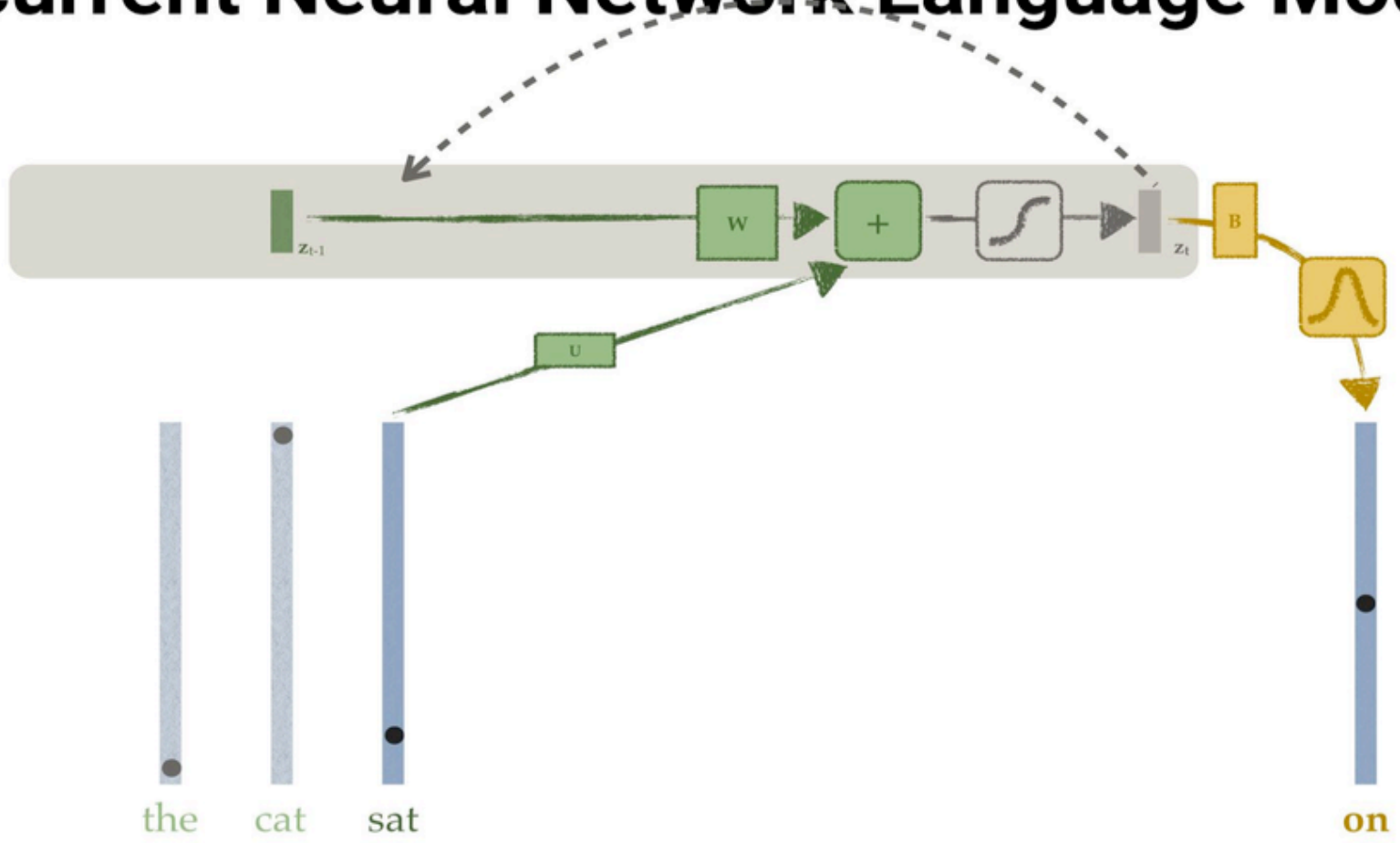
RNNs can handle






http://cs231n.stanford.edu/slides/

# Recurrent Neural Network Language Models

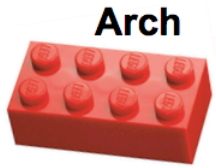


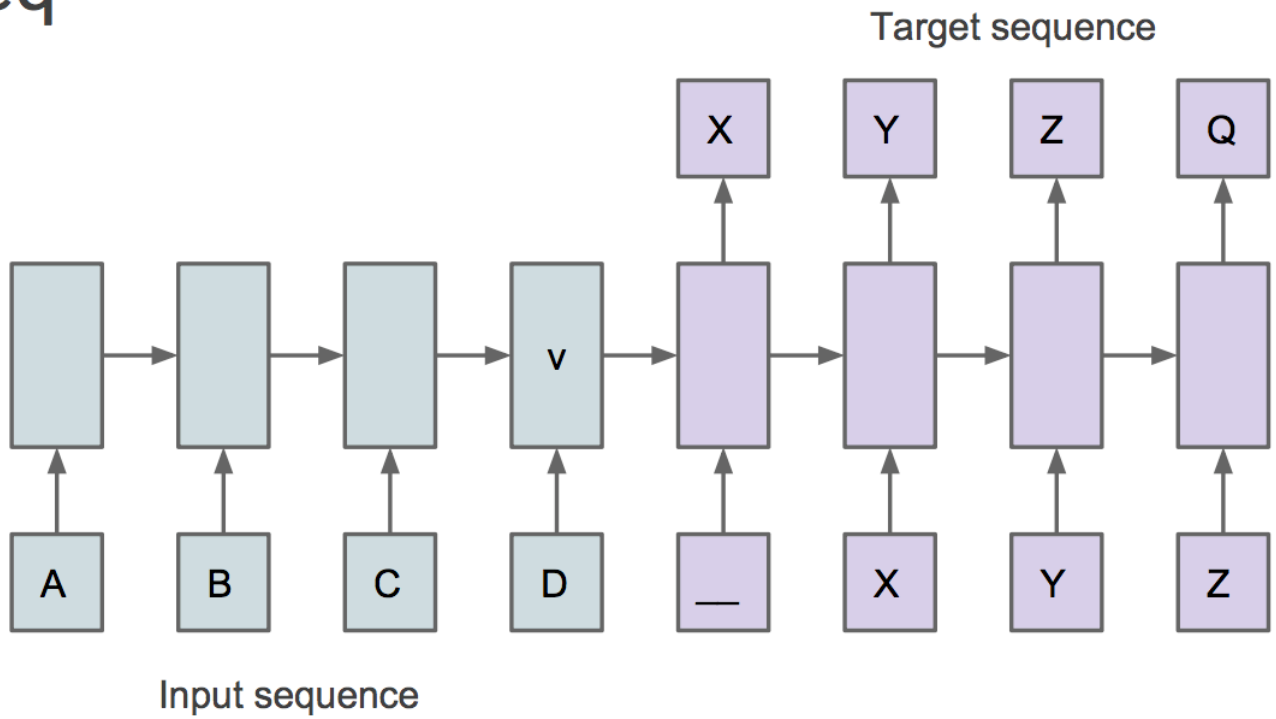



http://cs231n.stanford.edu/slides/

# e.g. For machine translation with 2RNN

## Seq2Seq



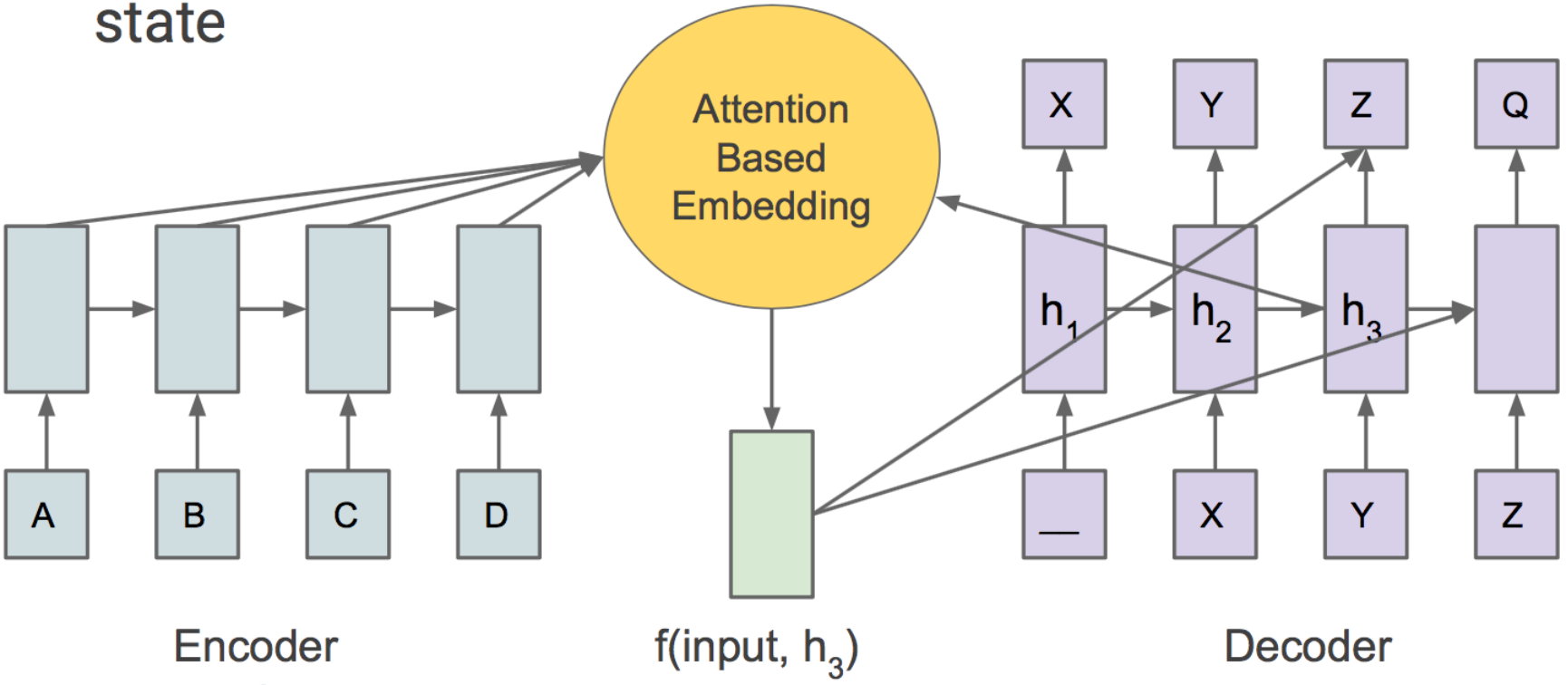




$$P(y_1, \ldots, y_{T'} | x_1, \ldots, x_T) = \prod_{t=1}^{T'} p(y_t | v, y_1, \ldots, y_{t-1})$$

**Attention Trick:**

# Seq2Seq with Attention

- Embedding used to predict output, and compute next hidden state



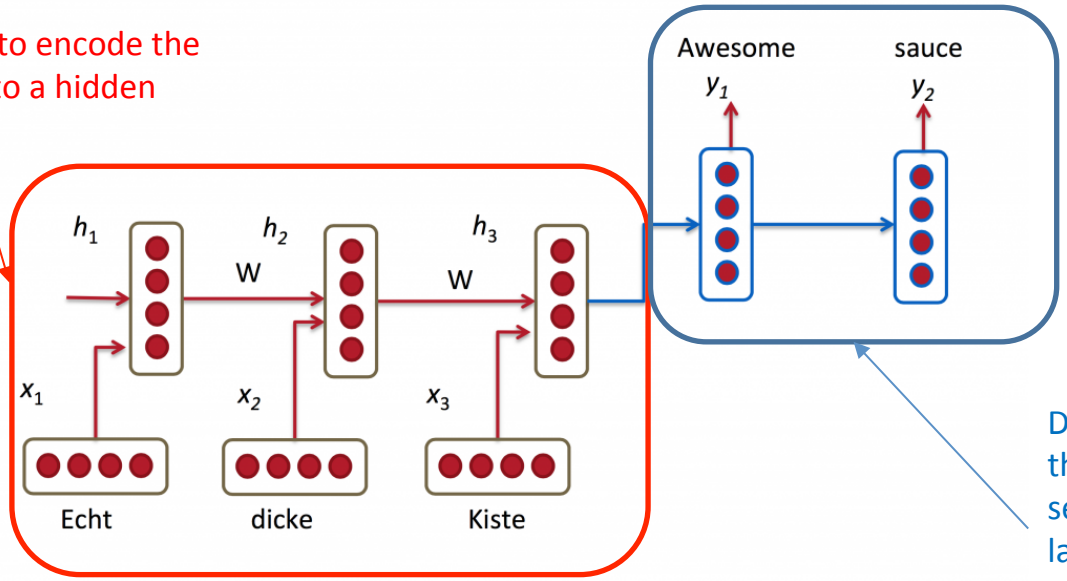



Adapt from From NIPS 2017 DL Trend Tutorial

# Seq2Seq for Machine Translation

In machine translation, the input is a sequence of words in source language, and the output is a sequence of words in target language.

Encoder: An RNN to encode the input sentence into a hidden state (feature)

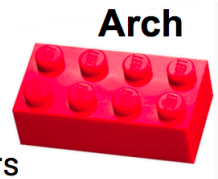



Decoder: An RNN with the hidden state of the sentence in source language as the input and output the translated sentence

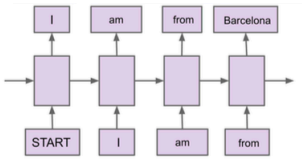
Encoder-decoder architecture for machine translation

Adapt from Professor Qiang Yang of HK UST

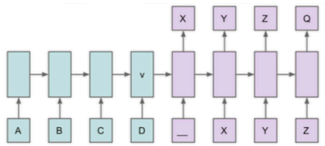
# Attention and Memory Toolbox



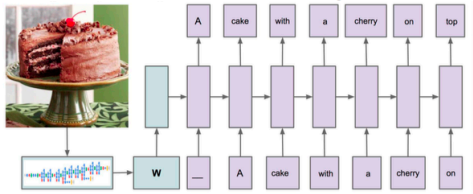

Sequence Prediction



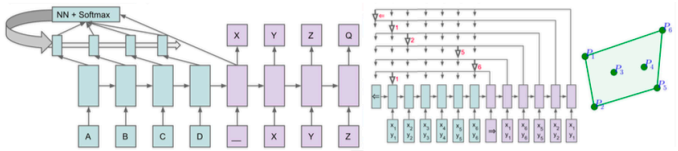

Seq2Seq



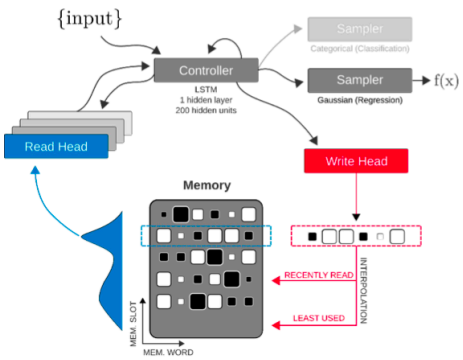

Multimodality

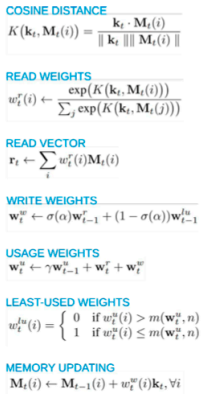



Attention/Pointers

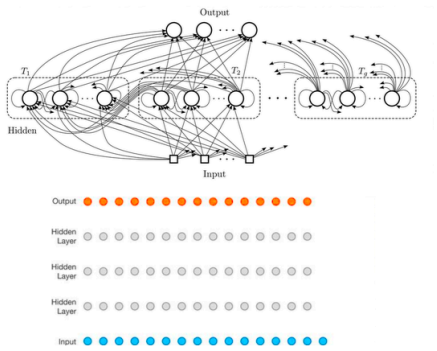



Read/Write memories

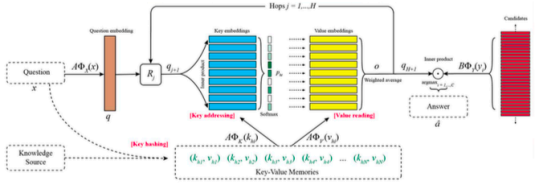


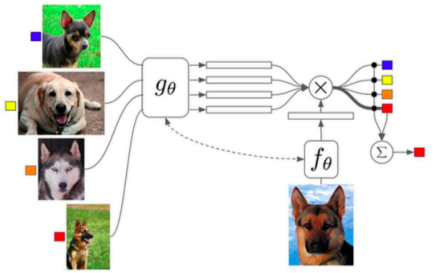




Temporal Hierarchies




Key,Value memories




Recurrent Architectures

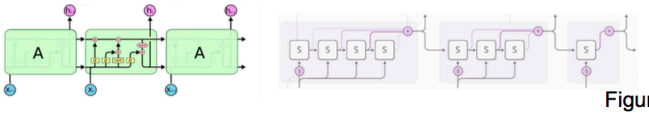



Figure credits: Jeff Dean, Chris Olah, Santoro et al 2016, Koutnik et al 2014, van den Oord et al 2016, Miller et al 2016, Vinyals et al 2016



Adapt from From NIPS 2017 DL Trend Tutorial

# Recent Trend (4): Neural Architectures with Memory

**Architectures:**

memory and multi-hop reasoning to perform AI tasks better

# Neural Architectures with Memory

- Antoine Bordes, Y-Lan Boureau, Jason Weston, **Learning End-to-End Goal-Oriented Dialog, ICLR 2017**
- Karol Kurach, Marcin Andrychowicz & Ilya Sutskever **Neural Random-Access Machines**, ICLR, 2016
- Emilio Parisotto & Ruslan Salakhutdinov **Neural Map: Structured Memory for Deep Reinforcement Learning**, ArXiv, 2017
- Oriol Vinyals,Meire Fortunato, Navdeep Jaitly **Pointer Networks**, ArXiv, 2017
- Jack W Rae et al., **Scaling Memory-Augmented Neural Networks with Sparse Reads and Writes, ArXiv 2016**
- Junhyuk Oh, Valliappa Chockalingam, Satinder Singh, Honglak Lee, **Control of Memory, Active Perception, and Action in Minecraft, ICML 2016**
- Wojciech Zaremba, Ilya Sutskever, **Reinforcement Learning Neural Turing Machines, ArXiv 2016**

# e.g. for Story Comprehension

Joe went to the kitchen. Fred went to the kitchen. Joe picked up the milk. Joe travelled to his office. Joe left the milk. Joe went to the bathroom.

Questions from Joe's angry mother:

Q1 : Where is Joe?

Q2 : Where is the milk now?

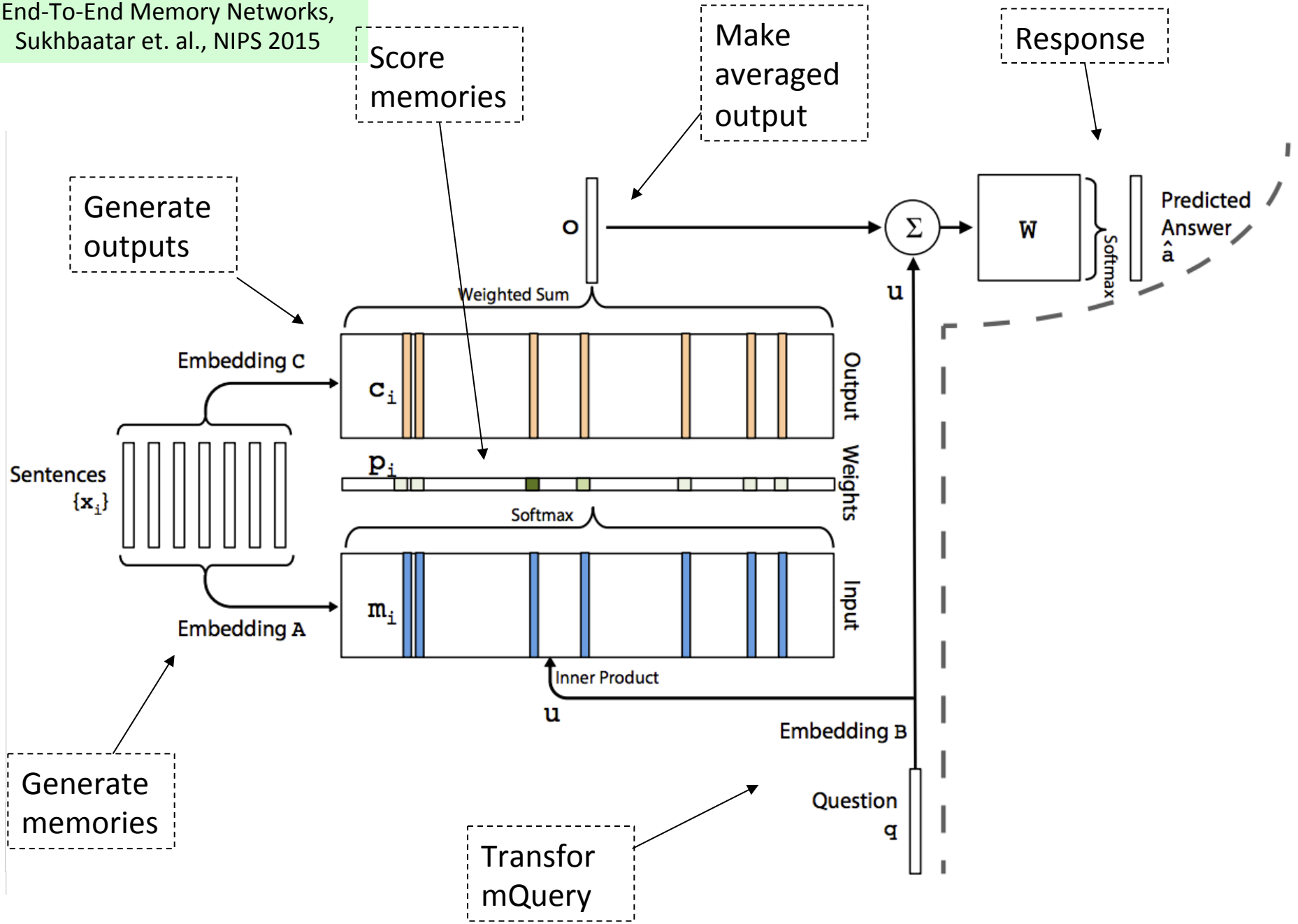
Q3 : Where was Joe before the office?

# Need external explicit memory for long-range reasoning

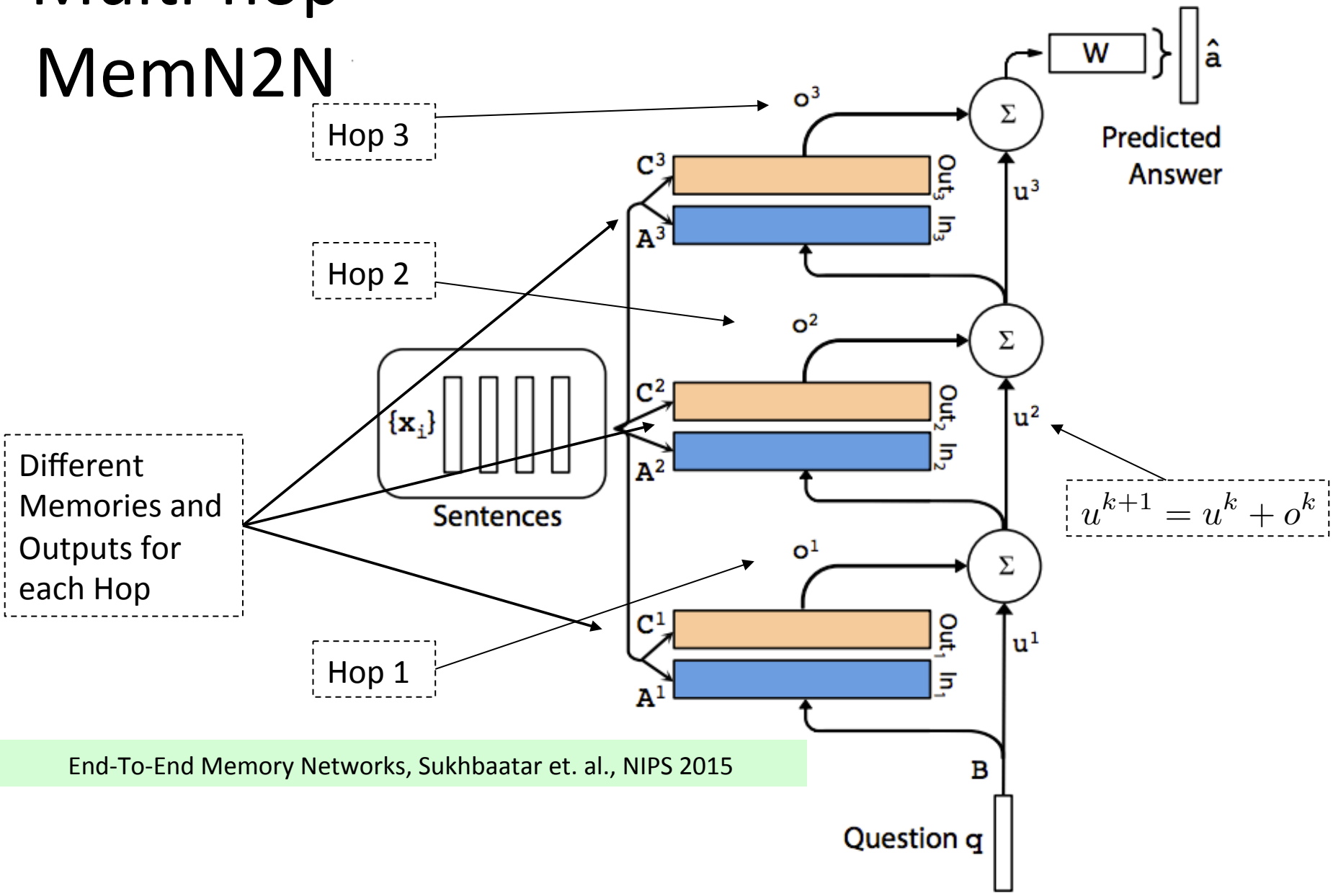
Deeper AI tasks require explicit memory and multi-hop reasoning over it

- RNNs have short memory
- Cannot increase memory without increasing number of parameters
- Need for compartmentalized memory
- Read/Write should be asynchronous

End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

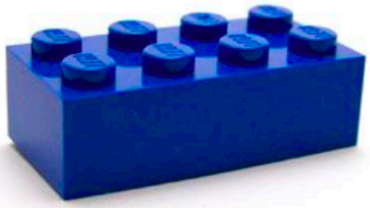


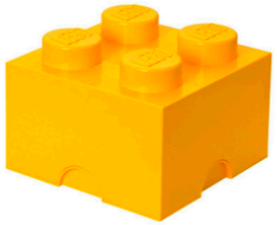
# Multi-hop MemN2N




End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

# Recent Trend (5): Tasks in the form of Sequence of Decisions / 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

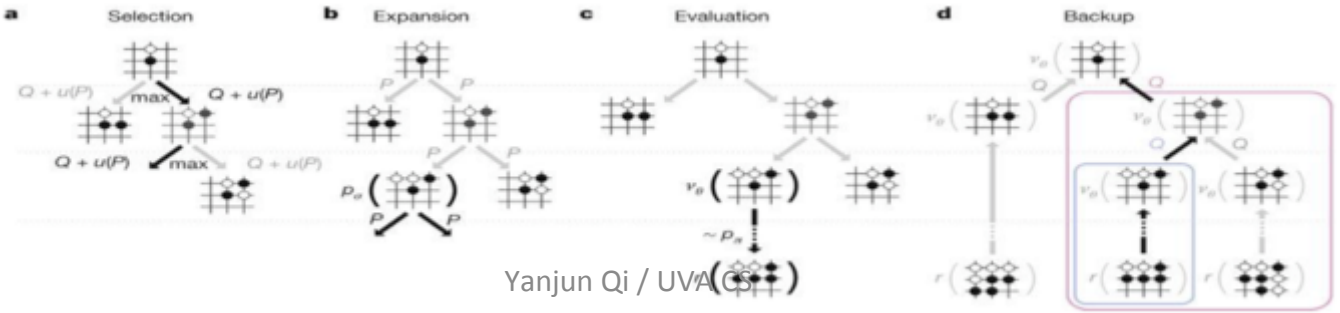
Adapt from From NIPS 2017 DL Trend Tutorial

# Task with Sequential Form

- Words, letters, strings, ..
- Speech, Audio, …
- Image, Video, Sensor touches, …
- Computer Programs , …

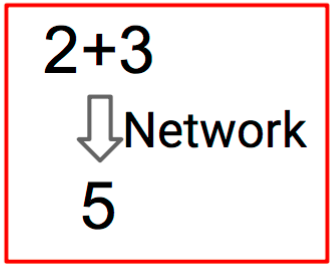
```
while (*d++ = *s++);
```

- Sequence decision making, e.g., games, RL

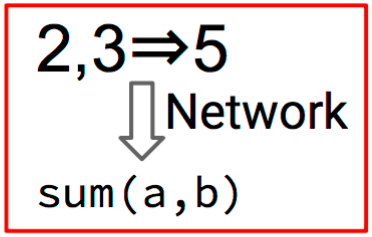


# Neural Program Induction - Research Landscape

- 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

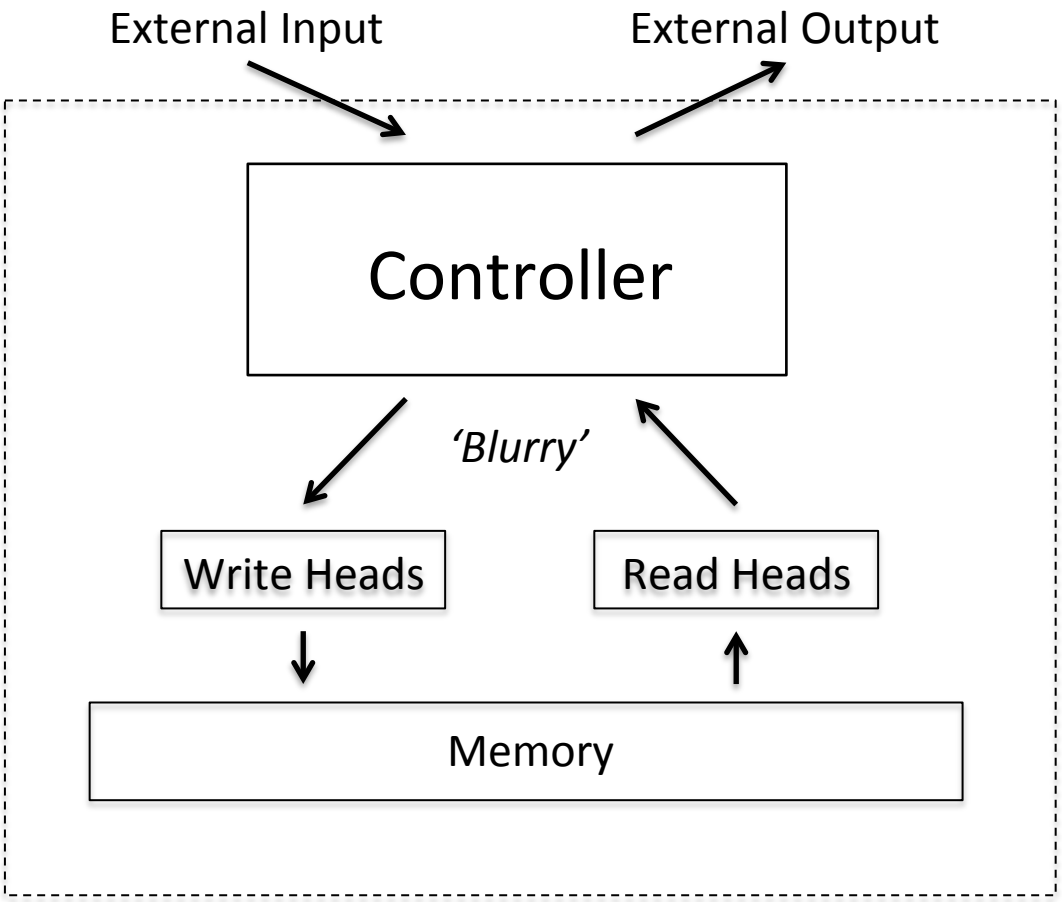



- Probabilistic programming with neural networks:
  - TerpreT, Edward, Picture



Adapt from From NIPS 2017 DL Trend Tutorial

# Neural Turing Machines




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

# Recent Trend (6): Learning to Optimize / Learning to Search DNN architecture

Inputs and Outputs

Losses

# Neural Architecture Search with Reinforcement Learning, ICLR17

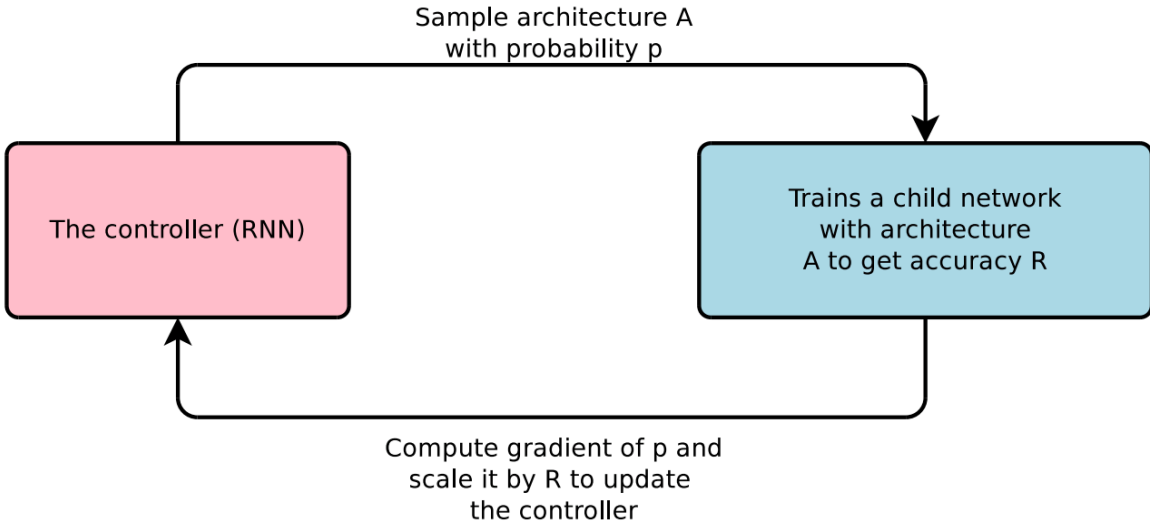
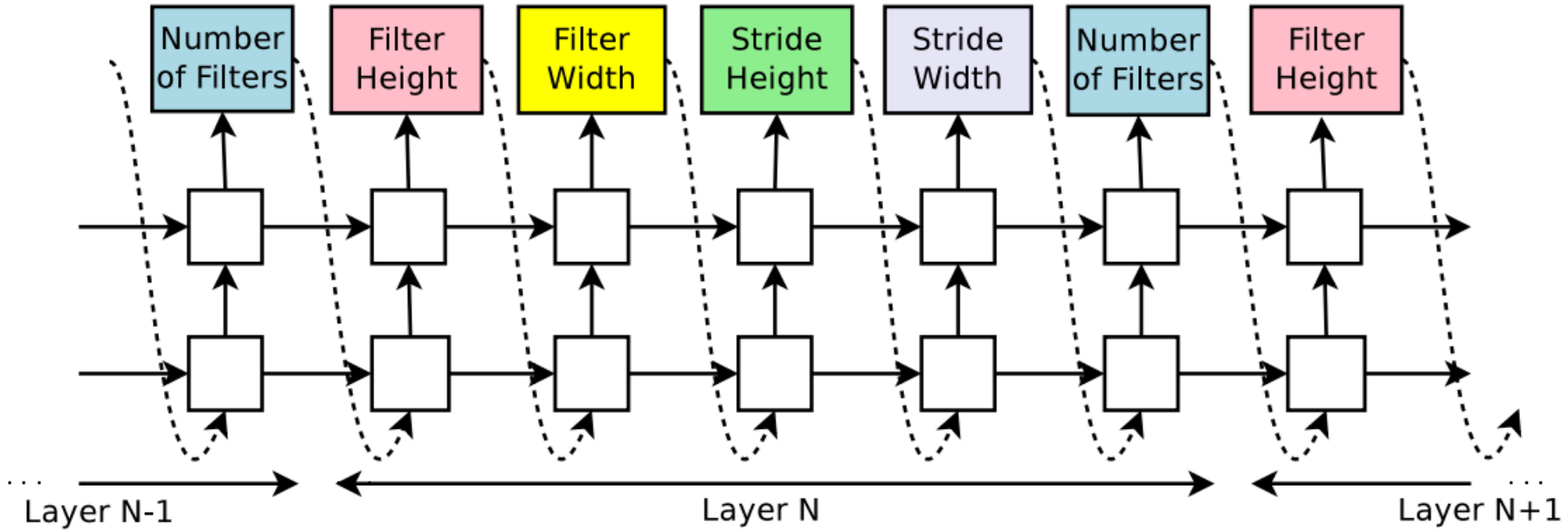



Figure 1: An overview of Neural Architecture Search.



Neural Optimizer Search with Reinforcement Learning, ICML17



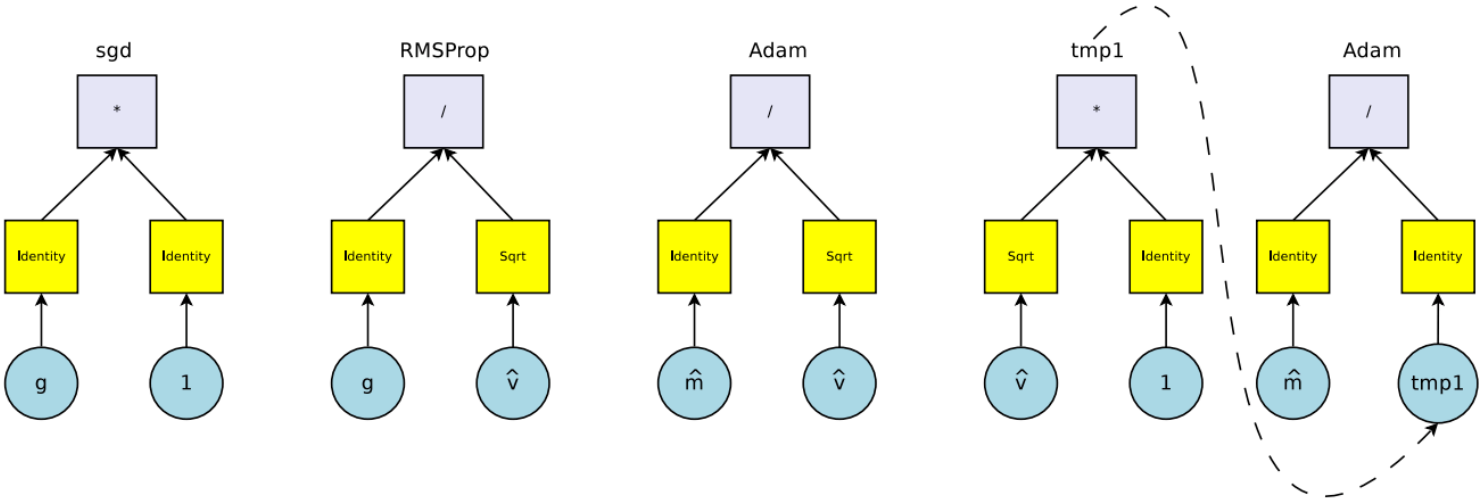



*Figure 2.* Computation graph of some commonly used optimizers: SGD, RMSProp, Adam. Here, we show the computation of Adam in 1 step and 2 steps. Blue boxes correspond to input primitives or temporary outputs, yellow boxes are unary functions and gray boxes represent binary functions. $g$ is the gradient,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

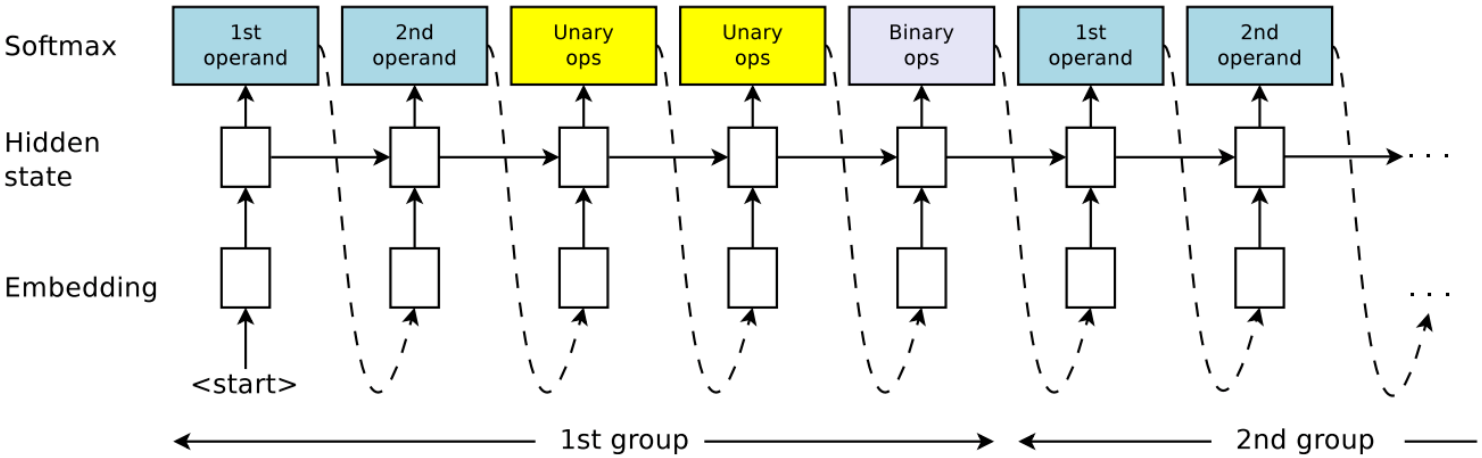


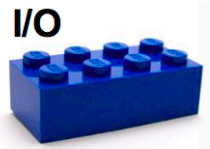

*Figure 3.* Overview of the controller RNN. The controller iteratively selects subsequences of length 5. It first selects the 1st and 2nd operands $op_1$ and $op_2$, then 2 unary functions $u_1$ and $u_2$ to apply to the operands and finally a binary function $b$ that combines the outputs of the unary 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.

# Recent Trend (7): Learning to Learn

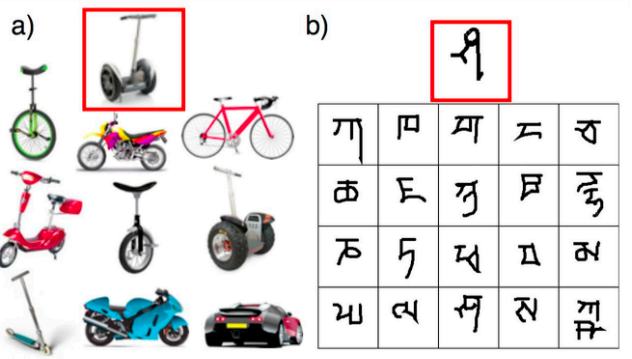
**Inputs and Outputs**

**Losses**

# 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*

**Image recognition** Mini-Imagenet dataset (Vinyals et al.’16)

Given 1 example of 5 classes: Classify new examples

**Reinforcement learning**

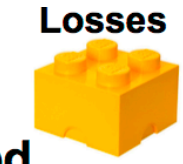
Given a small amount of experience Solve a new task

Chelsea Finn, UC Berkeley **How?** learn to learn many other tasks fig. from Duan et al. ’17

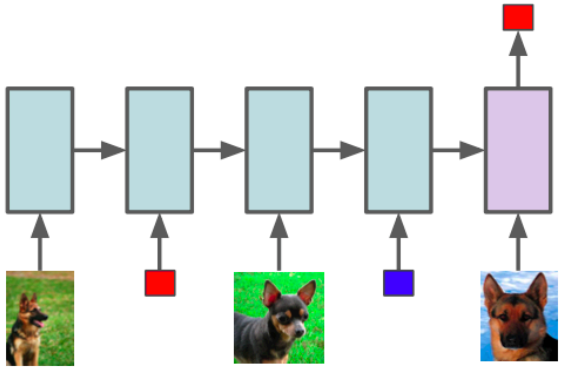


Adapt from From NIPS 2017 DL Trend Tutorial

# Learning to Learn

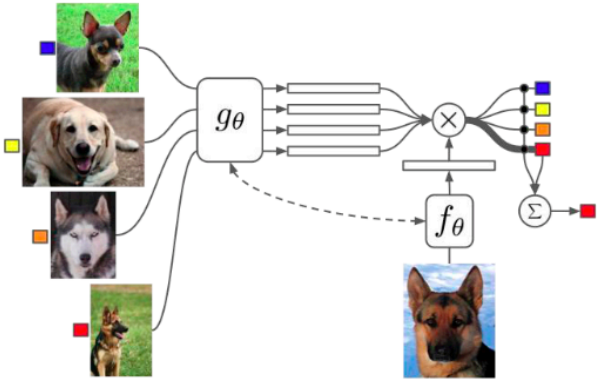



## Model Based



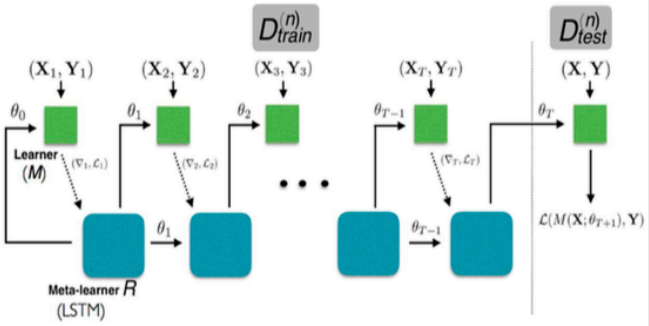
- Santoro et al. ’16
- Duan et al. ’17
- Wang et al. ‘17
- Munkhdalai & Yu ‘17
- Mishra et al. ‘17

## Metric Based




- Koch ’15
- Vinyals et al. ‘16
- Snell et al. ‘17
- Shyam et al. ‘17

## Optimization Based




- Schmidhuber ’87, ’92
- Bengio et al. ’90, ‘92
- Hochreiter et al. ’01
- Li & Malik ‘16
- Andrychowicz et al. ’16
- Ravi & Larochelle ‘17
- Finn et al ‘17



Adapt from From NIPS 2017 DL Trend Tutorial

# Recent Trend (8): Variants of Input, e.g., Graphs, Trees, Sets

**Inputs and Outputs**

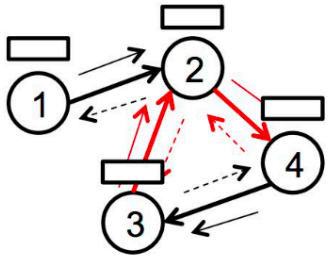
# Geometric Deep Learning on Graphs and Manifolds, NIPS 2017 Tutorial

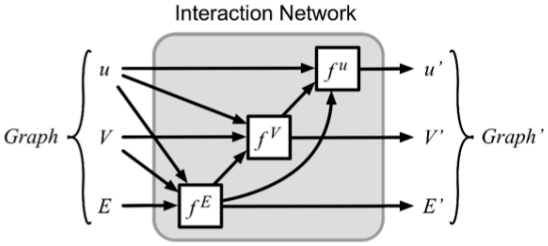
Graph Nets (GNs) are a class of models that:

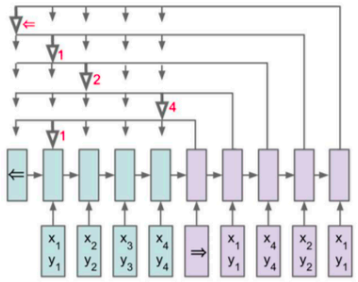
- Use graphs as inputs and/or outputs and/or latent representation
- Manipulate graph-structured representations
- Reflect relational structure
- Share model components across entities and relations

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)*



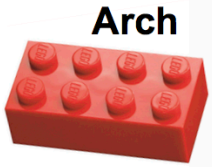



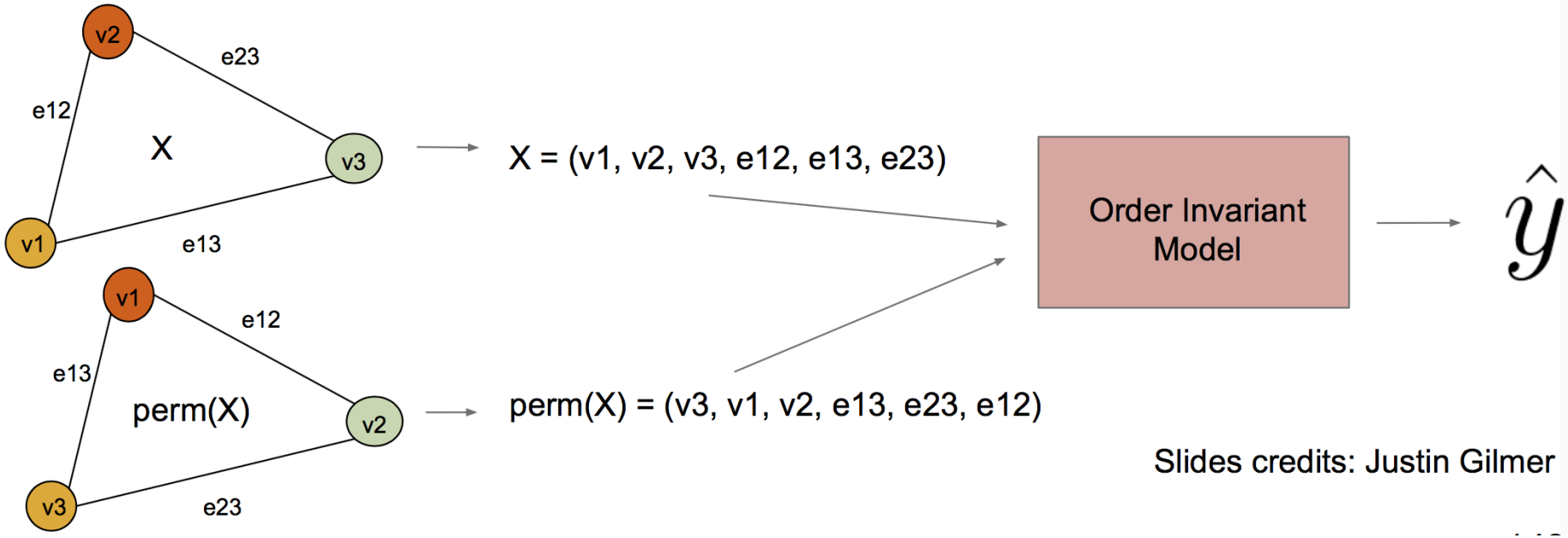


Adapt from From NIPS 2017 DL Trend Tutorial

# Inductive Bias for Graphs




- 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.

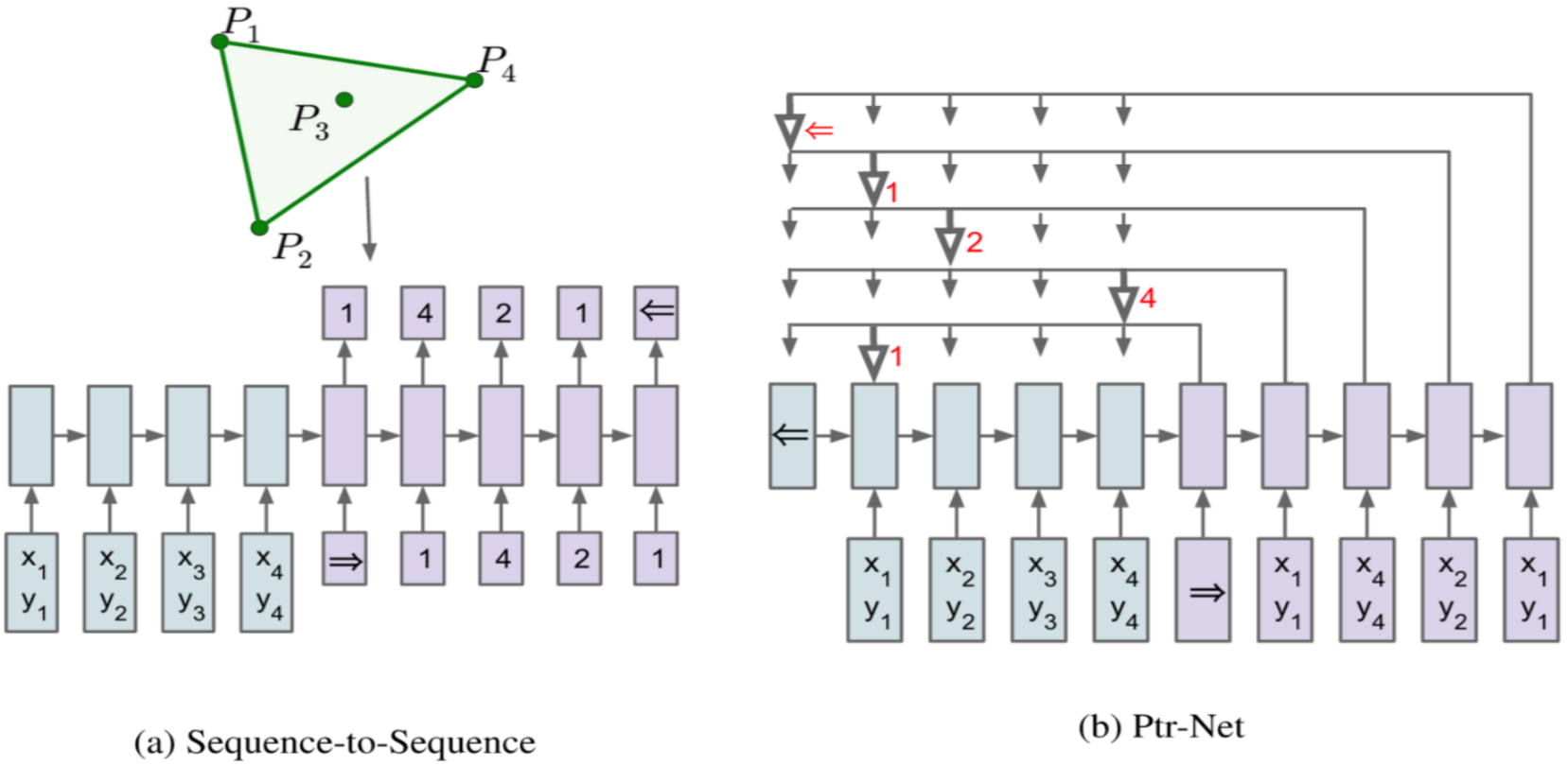



Slides credits: Justin Gilmer



Adapt from From NIPS 2017 DL Trend Tutorial

# Set input handled by the Pointer Networks- {NIPS 2015}




(a) Sequence-to-Sequence

(b) Ptr-Net

Figure 1: **(a)** *Sequence-to-Sequence* - An RNN (blue) processes the input sequence to create a code vector that is used to generate the output sequence (purple) using the probability chain rule and another RNN. The output dimensionality is fixed by the dimensionality of the problem and it is the same during training and inference [1]. **(b)** *Ptr-Net* - An encoding RNN converts the input sequence to a code (blue) that is fed to the generating network (purple). At each step, the generating network produces a vector that modulates a content-based attention mechanism over inputs ([5, 2]). The output of the attention mechanism is a softmax distribution with dictionary size equal to the length of the input.

# Recent Trend (9): Deep Generative Models: Autoregressive Kind

# 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

# 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



Adapt from From NIPS 2017 DL Trend Tutorial

# Autoregressive Models

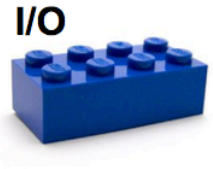
$$P(x;\theta) = \prod_{n=1}^{N} P(x_n | x_{<n}; \theta)$$

- Each factor can be parametrized by $\theta$, which can be shared.
- The variables can be arbitrarily ordered and grouped, as long as the ordering and grouping is consistent.



Adapt from From NIPS 2017 DL Trend Tutorial

# Modeling Text




The cat sat on the mat (word-level)

T h e _ c a t _ s a t _ o n _ t h e _ m a t (character-level)

The _ cat _ s a t _ on _ the _ m a t (mixed)

(byte level)

(bit level)

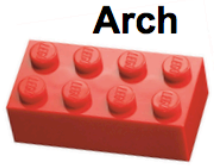
Shorter sequences and dependencies, semantically meaningful units, many UNK

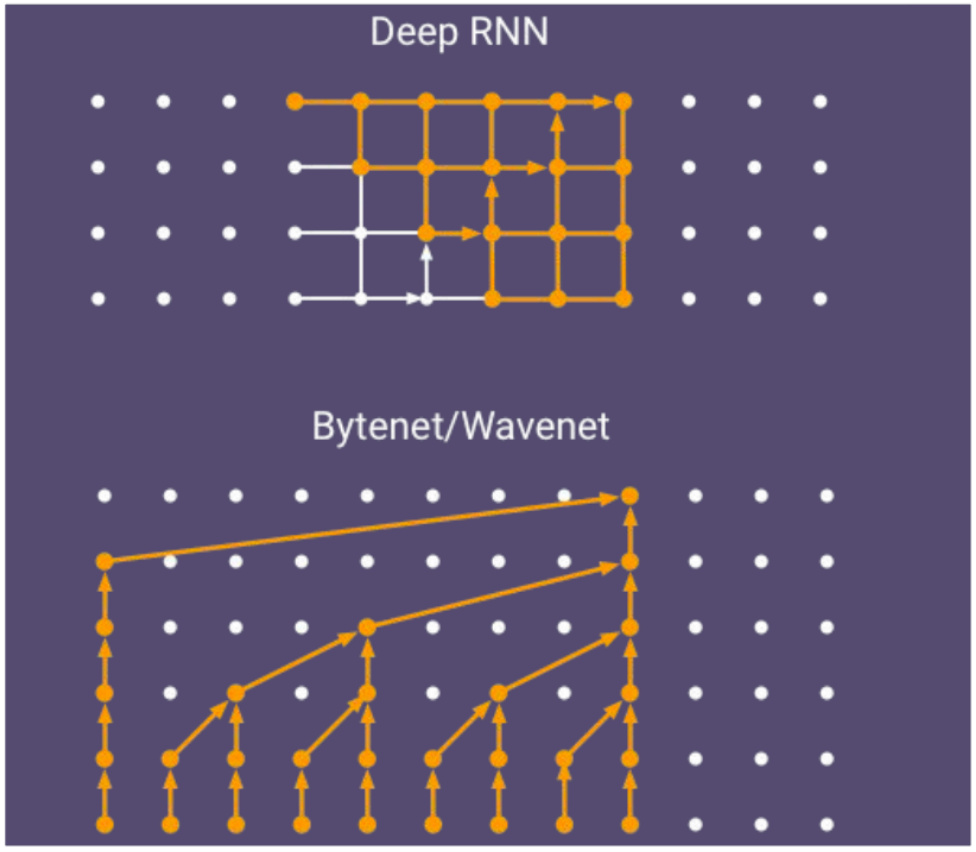
Long sequences and dependencies, semantically not meaningful units, no UNK



Adapt from From NIPS 2017 DL Trend Tutorial

# Recurrent versus Causal Convolutional Nets




- The architecture is parallelizable along the time dimension (during training or scoring)
- Easy access to many states from the past



Adapt from From NIPS 2017 DL Trend Tutorial

# Recent Trend (10): Generative Adversarial Networks (GAN)

**Architectures:**

**Losses**

10 BREAKTHROUGH TECHNOLOGIES 2018

MIT Technology Review

## Dueling Neural Networks

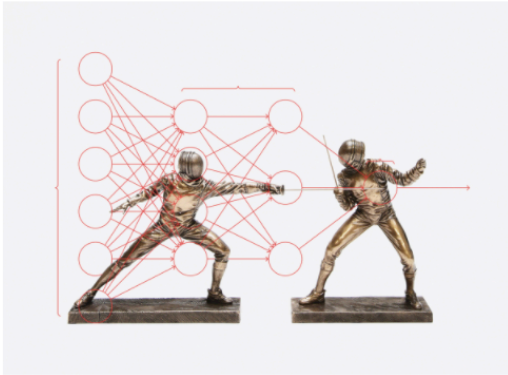


ILLUSTRATION BY DEREK BRAHNEY | DIAGRAM COURTESY OF MICHAEL NIELSEN, "NEURAL NETWORKS AND DEEP LEARNING", DETERMINATION PRESS, 2015

# Adversarial Nets Framework

$D$(x) tries to be near 1

Differentiable function $D$

$x$ sampled from data

$D$ tries to make $D(G(z))$ near 0, $G$ tries to make $D(G(z))$ near 1

$D$

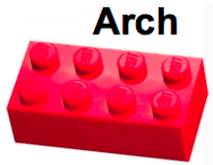
$x$ sampled from model

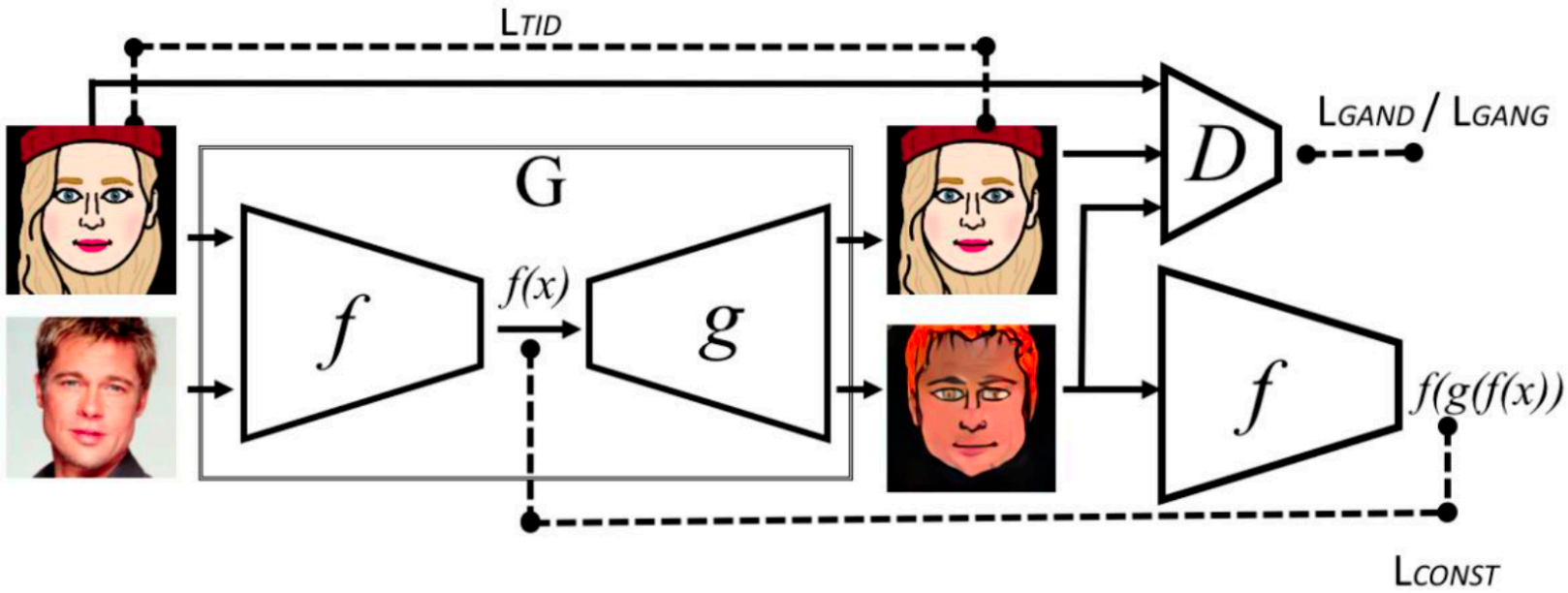
Differentiable function $G$

Input noise $z$

(Goodfellow 2016)

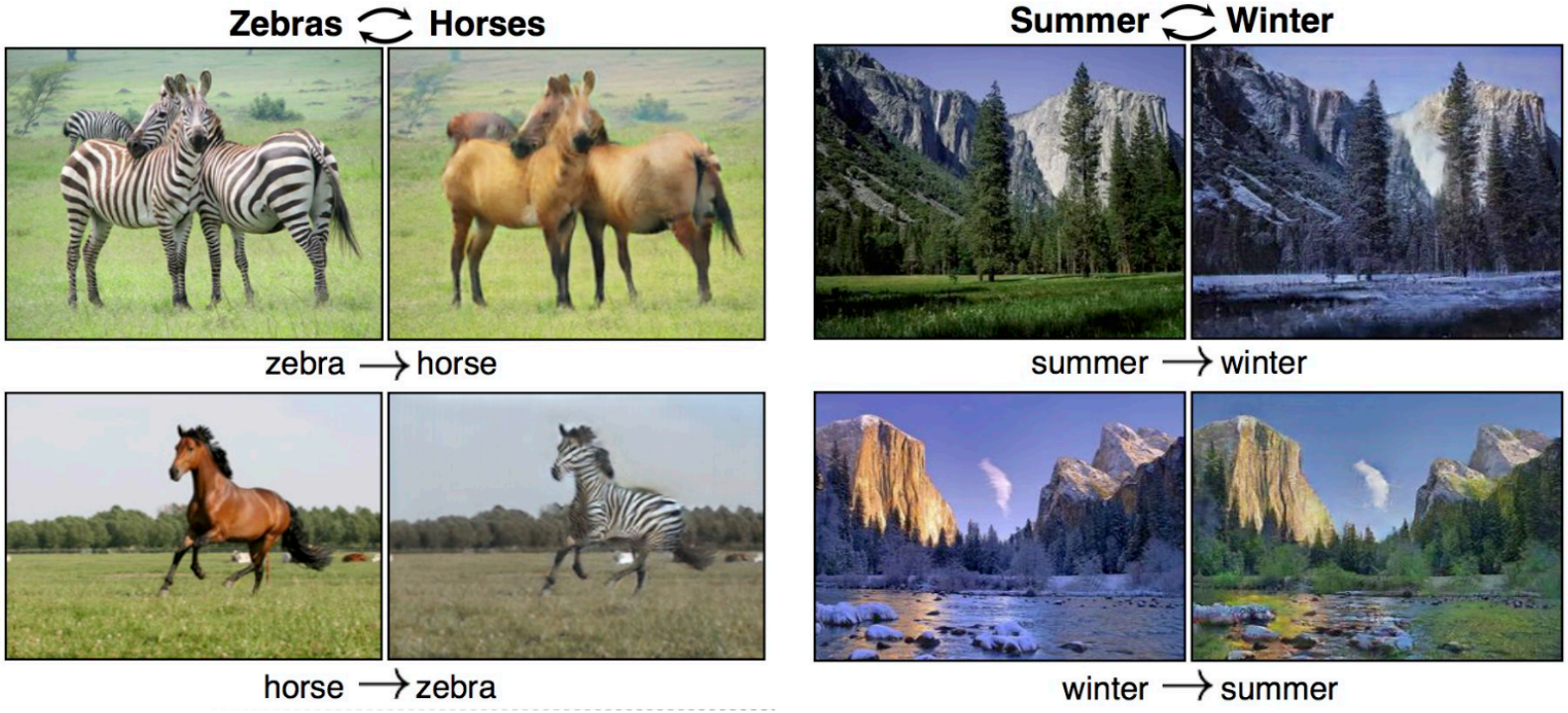
# Unsupervised cross-domain image generation




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

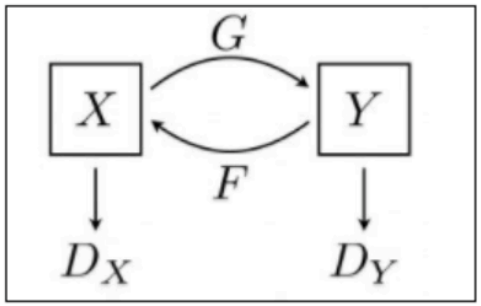
# CycleGAN




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

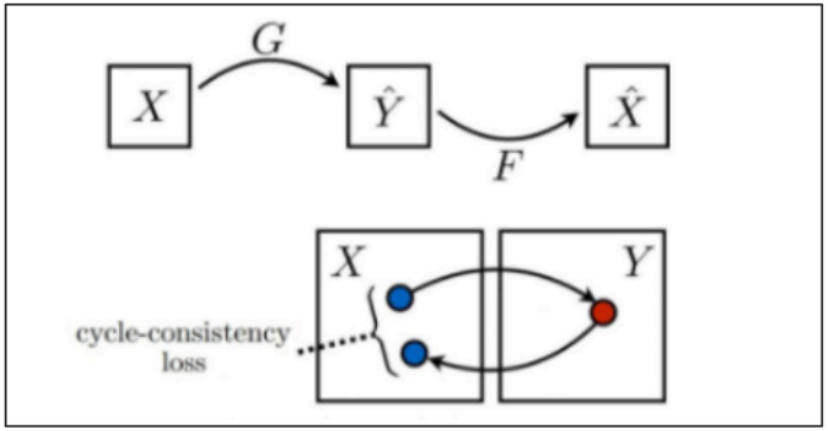
This paper captures special characteristics of one image collection and figures out how these characteristics could be translated into the other image collection, all in the absence of any paired training examples. CycleGANs method can also be applied in variety of applications such as Collection Style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

# CycleGAN




*Source: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*

The way CycleGANs are able to learn such great translations without having explicit X/Y training images involves introducing the idea of a full translation cycle to determine how good the entire translation system is, thus improving both generators at the same time.



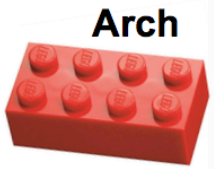

https://datahub.packtpub.com/deep-learning/2017-generative-adversarial-networks-gans-research-milestones/

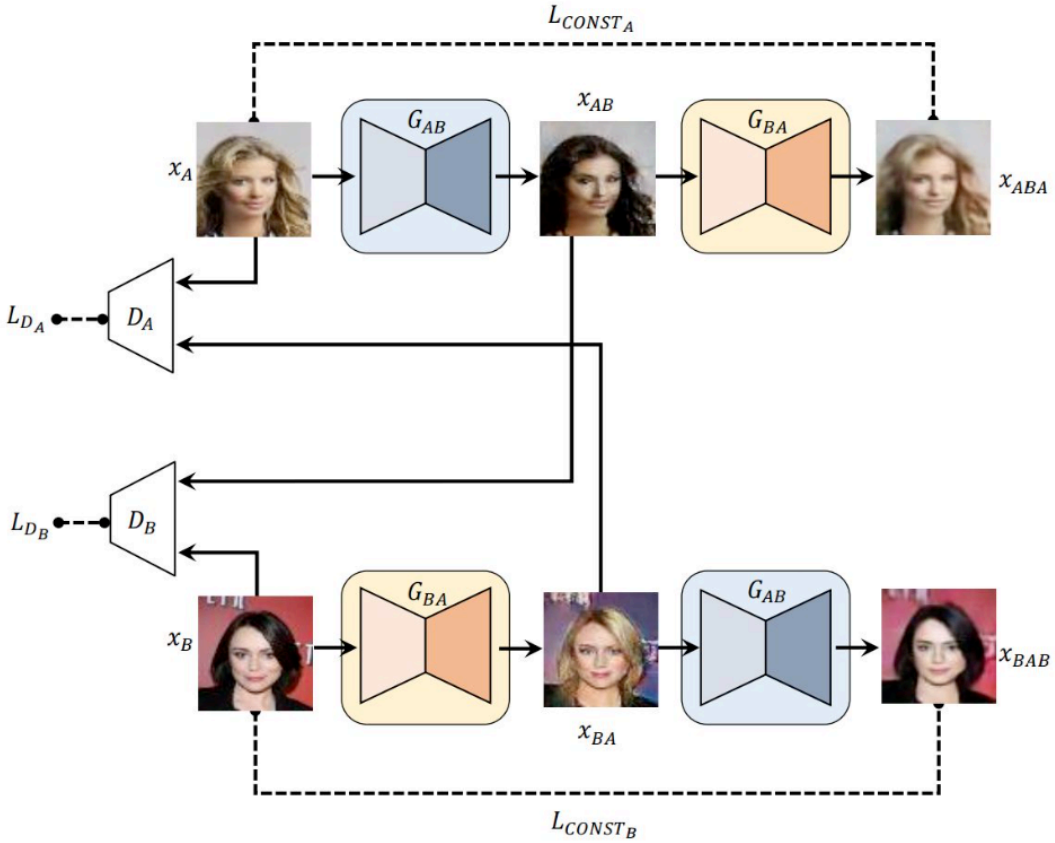
Source: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Currently, the applications of CycleGANs can be seen in Image-to-Image translation and video translations. For example they can be seen used in Animal Transfiguration, Turning portrait faces into doll faces, and so on. Further ahead, we could potentially see its implementations in audio, text, etc., would help us in generating new data for training.

# Cross-domain with DiscoGAN

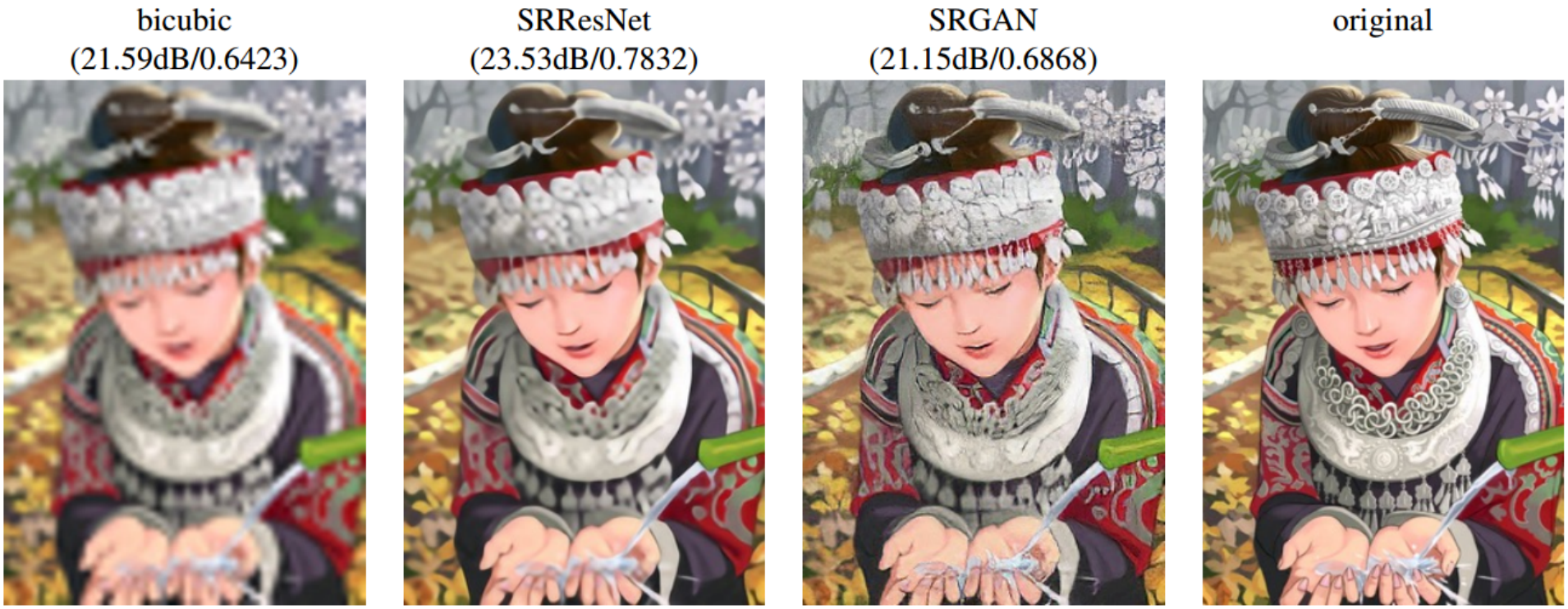
DiscoGAN




1. Kim et al. "Learning to Discover Cross-Domain Relations with Generative Adversarial Networks". In ICML, 2017.

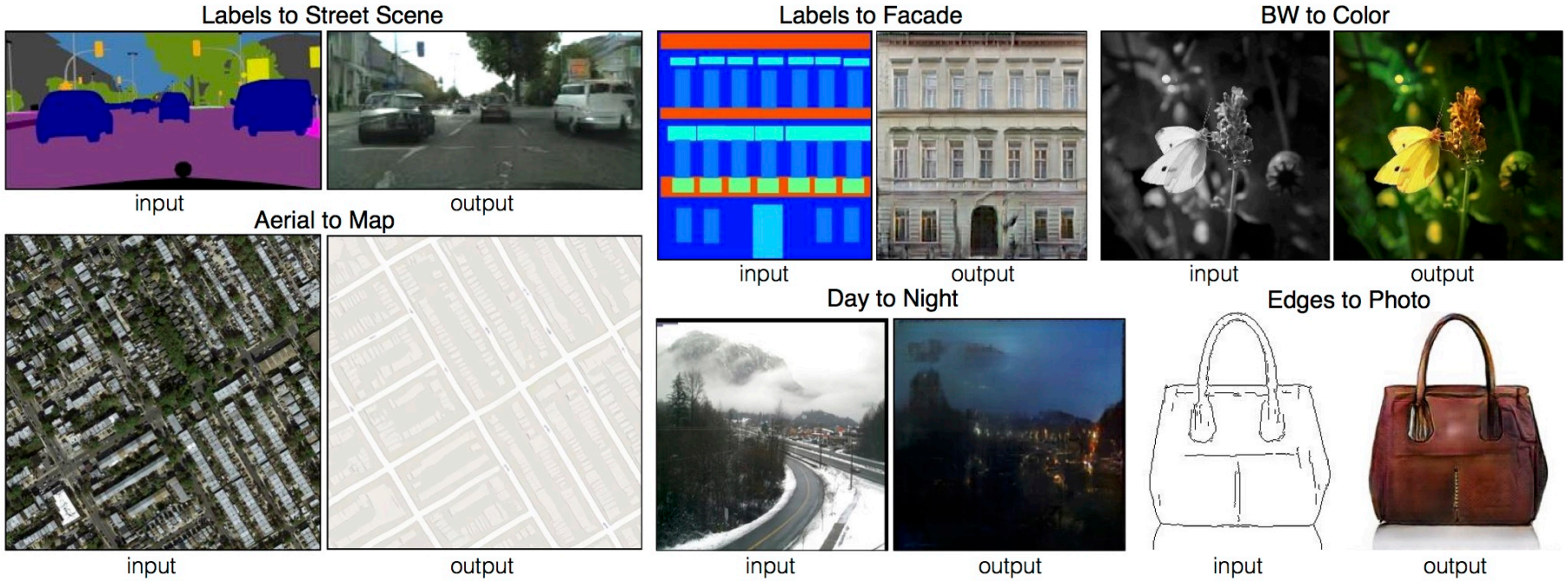
# Image Super-Resolution




- Conditional on low-resolution input image

[Ledig et al. CVPR 2017]

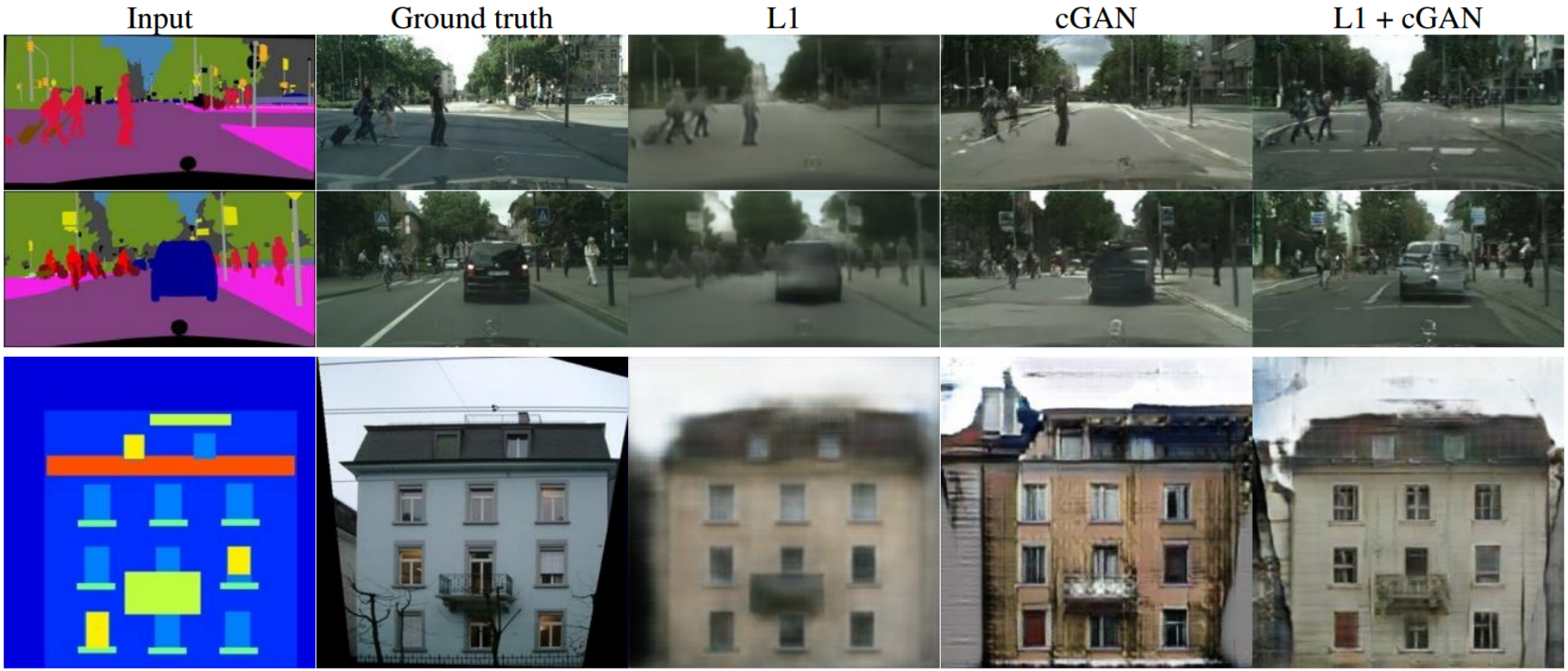
# Image-to-Image Translation




- Conditioned on an image of different modality
- No need to specify the loss function

[Isola et al. CVPR 2017]

# Label2Image




[Isola et al. CVPR 2017]

# Edges2Image


[Isola et al. CVPR 2017]

# 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

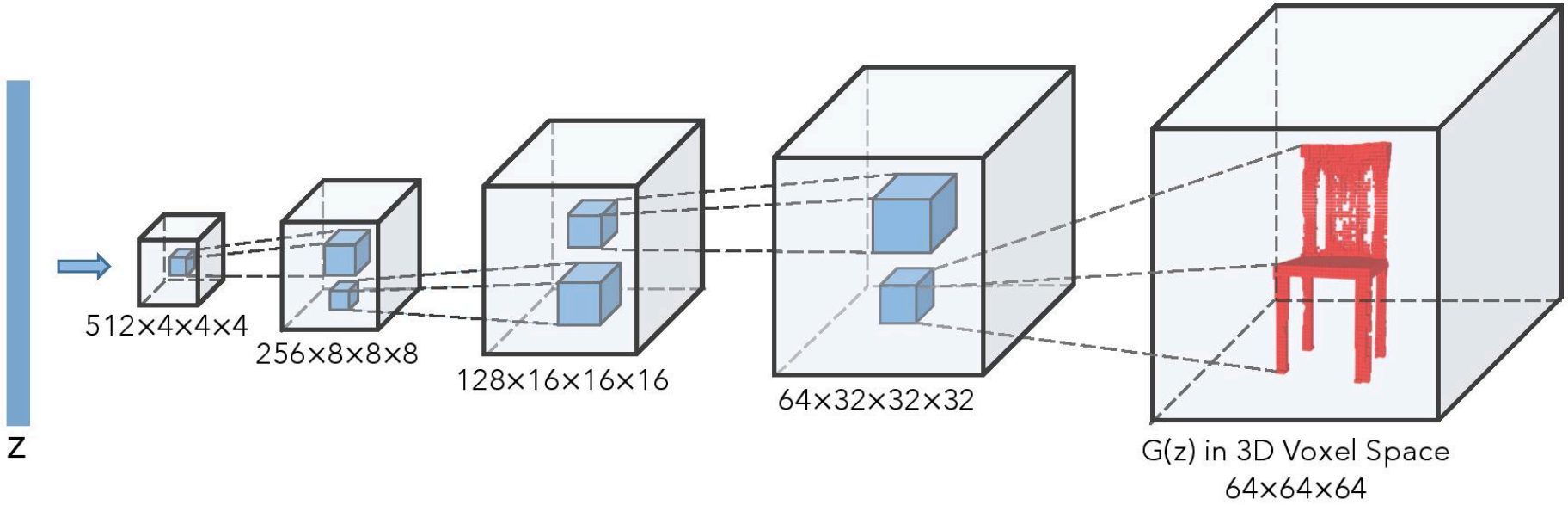
[Reed et al. ICML 2016]

# Plug & Play Generative Networks



[Nguyen et al. 2016]

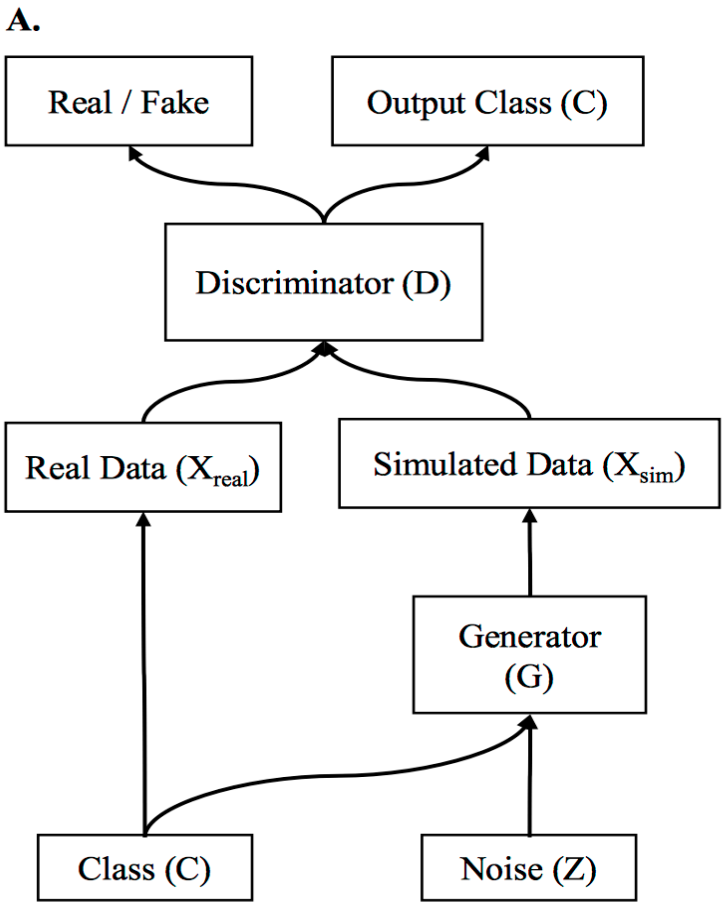
# Shape modeling using 3D Generative Adversarial Network

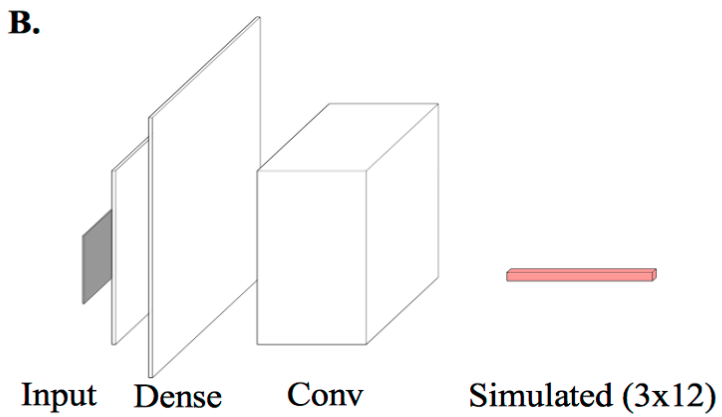


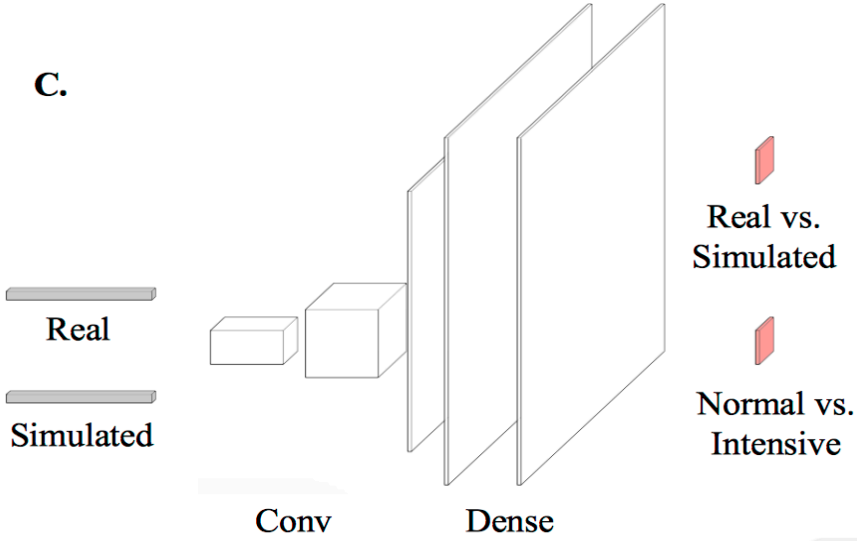

[Wu et al. NIPS 2016]

# SPRINT_GAN: Privacy Preserving GAN

We train deep neural networks that generate synthetic subjects closely resembling study participants. Using the SPRINT trial as an example, we show that machine-learning models built from simulated participants generalize to the original dataset. We incorporate differential privacy, which offers strong guarantees on the likelihood that a subject could be identified as a




Privacy-preserving generative deep neural networks support clinical data sharing http://www.biorxiv.org/content/early/...

# Progressive GAN


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

# Recent Trend (11): Deep Reinforcement Learning

**10 Breakthrough Technologies**

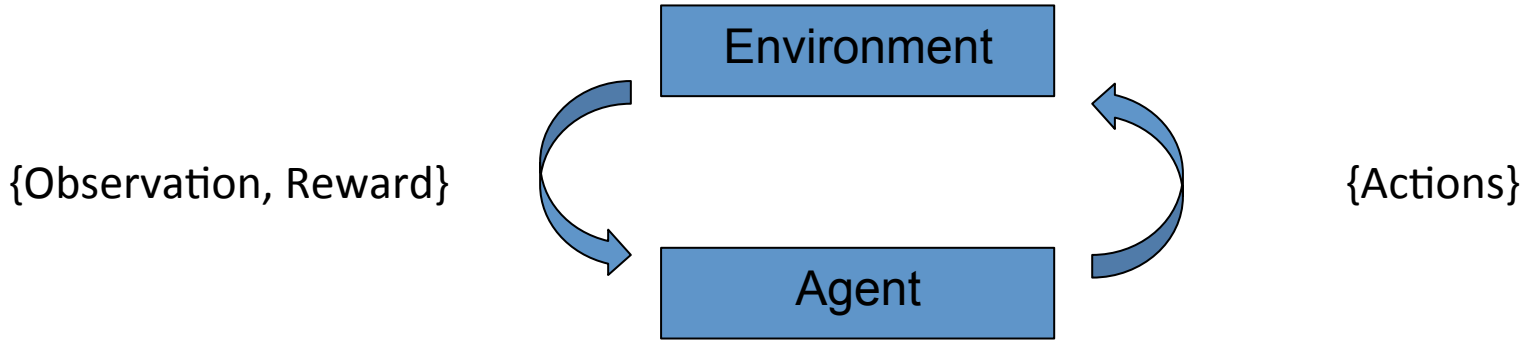
2017

MIT Technology Review

These 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.

# Reinforcement Learning (RL)

- What's Reinforcement Learning?

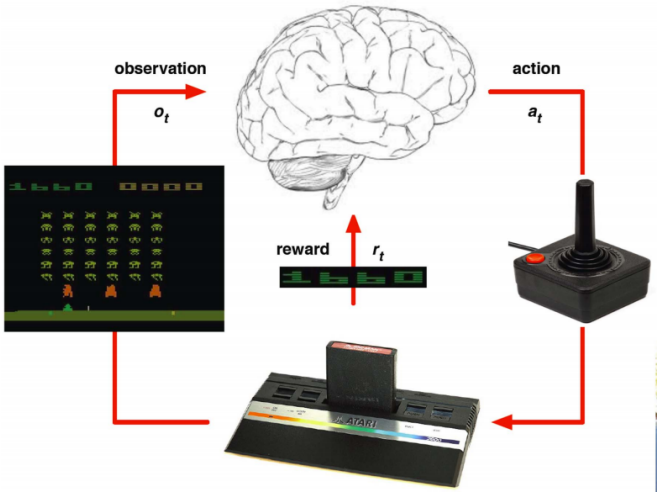



- Agent interacts with an environment and learns by maximizing a scalar reward signal
- No labels or any other supervision signal.
- Previously suffering from hand-craft states or representation.

Adapt from Professor Qiang Yang of HK UST

# Deep Reinforcement Learning

- Human

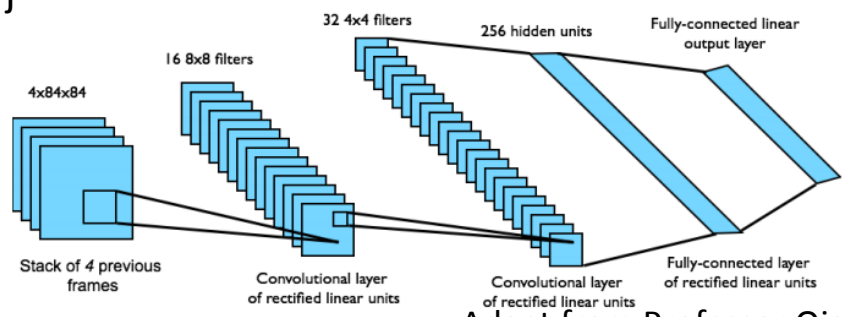



- So what's **DEEP** RL?

Environment

{Raw Observation, Reward}

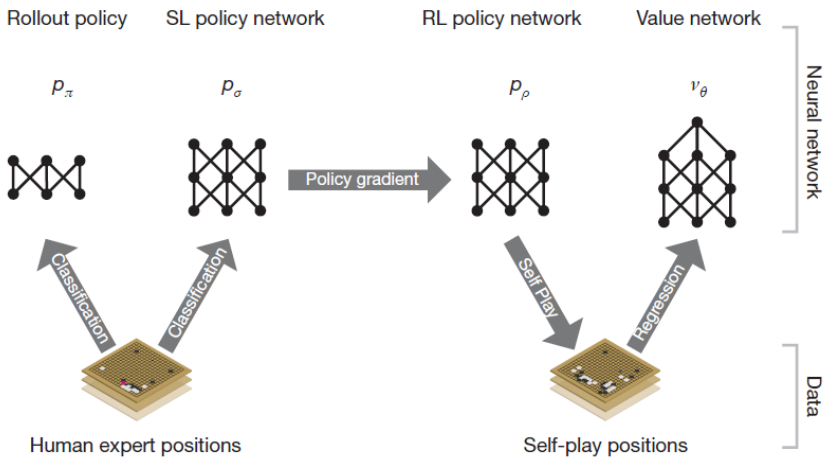
{Actions}




Adapt from Professor Qiang Yang of HK UST

# 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

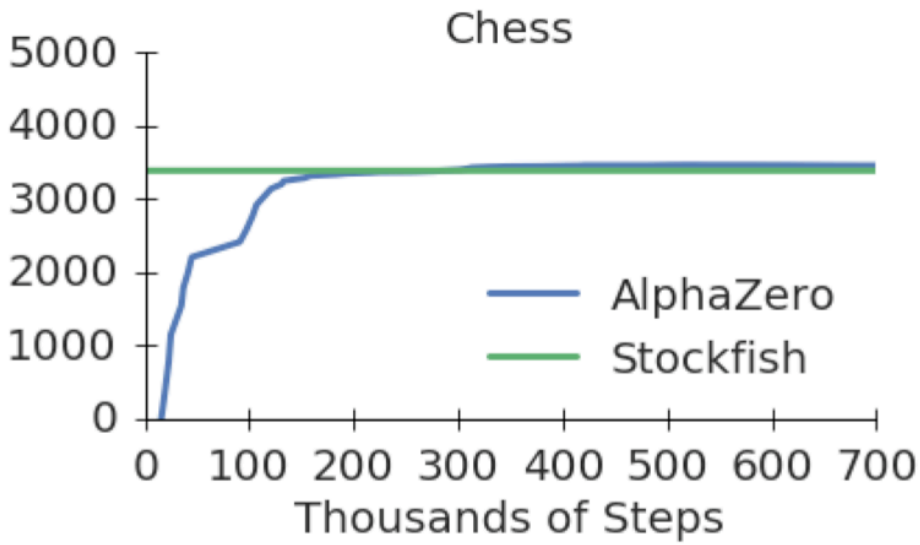
Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search." *Nature* 529.7587 (2016): 484-489.

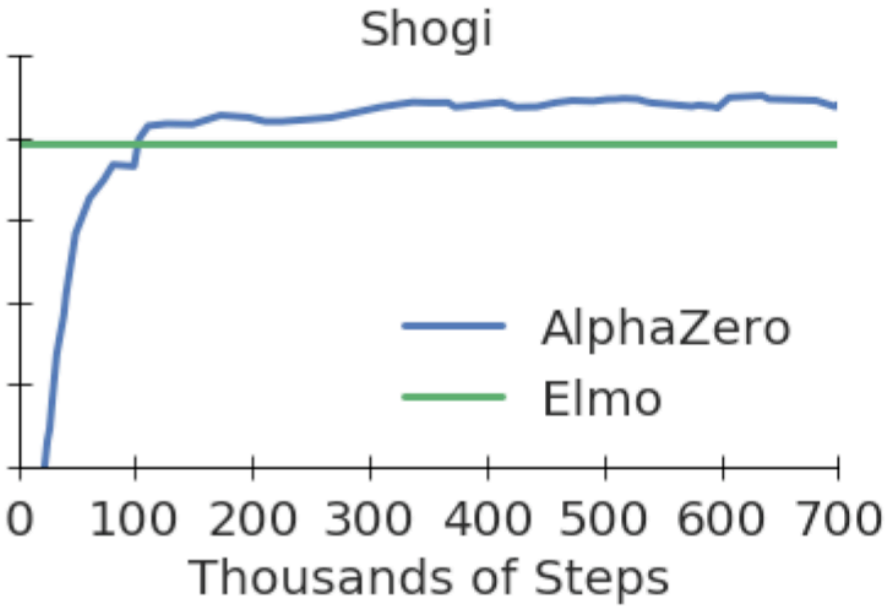
AlphaGo {Fan, Lee, Master} × AlphaGo Zero:

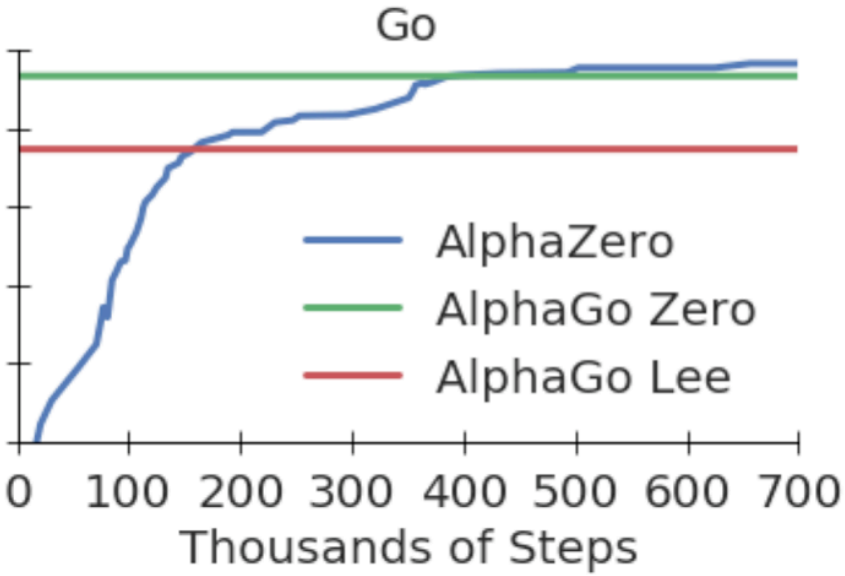
- supervised learning from human expert positions × from scratch by self-play reinforcement learning ("tabula rasa")
- additional (auxialiary) input features × only the black and white stones from the board as input features
- separate policy and value networks × single neural network
- tree search using also Monte Carlo rollouts × simpler tree search using only the single neural network to both evaluate positions and sample moves
- (AlphaGo Lee) distributed machines + 48 tensor processing units (TPUs) × single machines + 4 TPUs
- (AlphaGo Lee) several months of training time × 72 h of training time (outperforming AlphaGo Lee after 36 h)



Silver, David et al. (2017b). "Mastering the Game of Go without Human Knowledge". In: Nature 550.7676, pp. 354–359.






Silver, David et al. (2017b). "Mastering the Game of Go without Human Knowledge". In: Nature 550.7676, pp. 354–359.

# Recent Trend (12): Robustness / Trustworthiness / Understand / Verify / Test / Evade / Detect Bias / Protect DNN

**Validation**

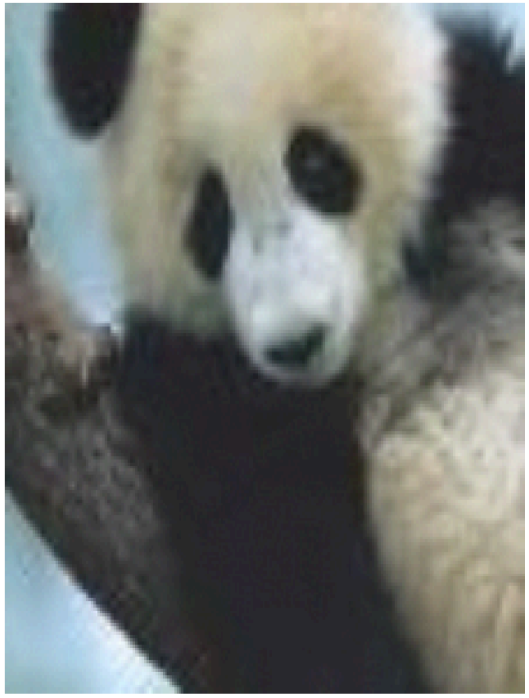
# Evade DNN, e.g. Adversarial Examples (AE)

“panda”

+

$0.007 \times [noise]$

=



“gibbon”

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

# A few references for AE


- I.J. Goodfellow, J. Shlens, C. Szegedy, “Explaining and Harnessing Adversarial Examples.” ArXiv e-prints, December 2014.
- B. Biggio, F. Roli, “Wild patterns: Ten years after the rise of adversarial machine learning.” arXiv preprint arXiv:1712.03141, 2017.
- N. Papernot, P. McDaniel, A. Sinha, M. Wellman, “Towards the science of security and privacy in machine learning.” IEEE European Symposium on Security and Privacy, 2018.
- N. Papernot, P. Mcdaniel, I.J. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples” arXiv preprint arXiv:1605.07277, 2016.
- C. Szegedy, W. Zaremba, I. Sutskever, J Bruna, D. Erhan, I.J. Goodfellow, R. Fergus. “Intriguing properties of neural networks.” ICLR, abs/1312.6199, 2014.
- Kurakin A, Goodfellow I, Bengio S. Adversarial examples in the physical world. arXiv preprint arXiv: 1607.02533. 2016 Jul 8.
- A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu, “Towards Deep Learning Models Resistant to Adversarial Attacks”. February 2018.

# Bias in DNN: e.g. Men Also Like Shopping: Reducing Gender Bias Amplification using Corpus-level Constraints, EMNLP 2017

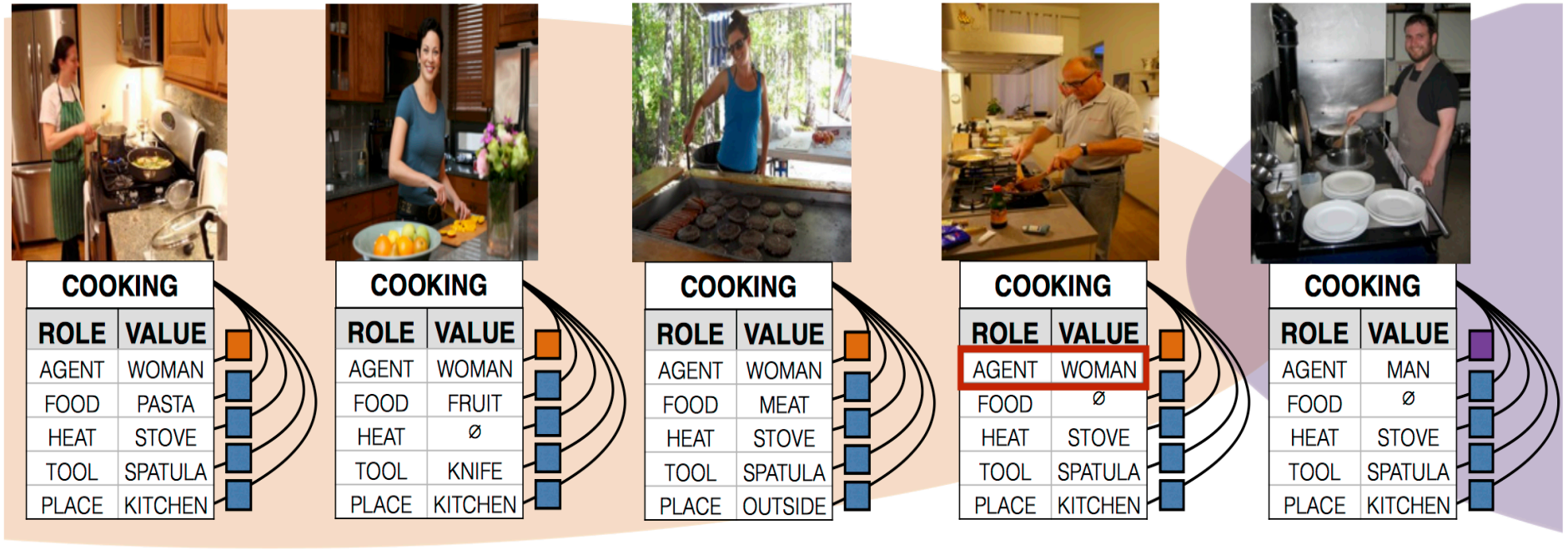



Figure 1: Five example images from the imSitu visual semantic role labeling (vSRL) dataset. Each image is paired with a table describing a situation: the verb, `cooking`, its semantic roles, i.e `agent`, and noun values filling that role, i.e. `woman`. In the imSitu training set, 33% of `cooking` images have `man`

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

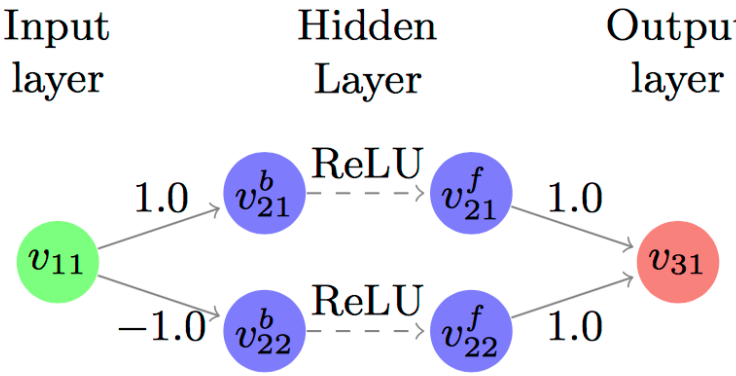



Table 3: Local adversarial robustness tests. All times are in seconds.

| | $\delta = 0.1$ | | $\delta = 0.075$ | | $\delta = 0.05$ | | $\delta = 0.025$ | | $\delta = 0.01$ | | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Result | Time | Result | Time | Result | Time | Result | Time | Result | Time | 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 |

# Today Recap

- Deep Learning
  - Basics
  - History
  - Why is this a breakthrough ?
  - Recent trends
  - (Many more exciting trends not covered here!)