CORNELL TECH

Deep Learning Clinic

(DLC)

Lecture 4
A Brief Introduction to Deep Learning

Jin Sun

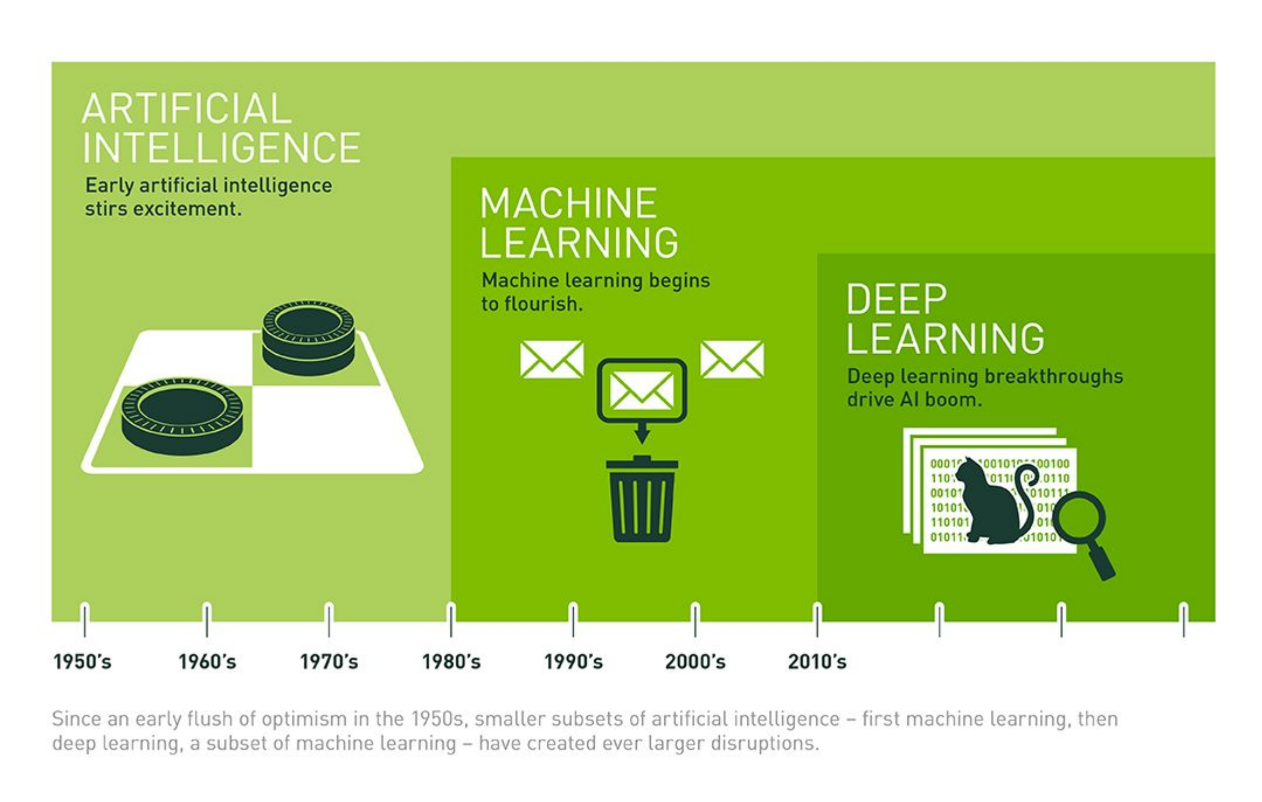
10/19/2018

Today

- **Overview**
- Basic Feedforward Networks and Core Concepts
  - Optimization
  - Regularization
- Convolutional Networks
- Recurrent Networks
- Generative Models
- Research Frontiers

Slides adapted from Ian Goodfellow.

Overview



https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

What is Deep Learning

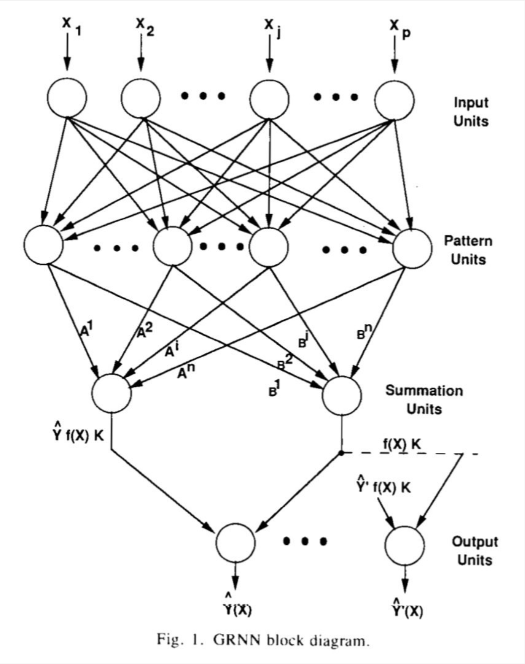
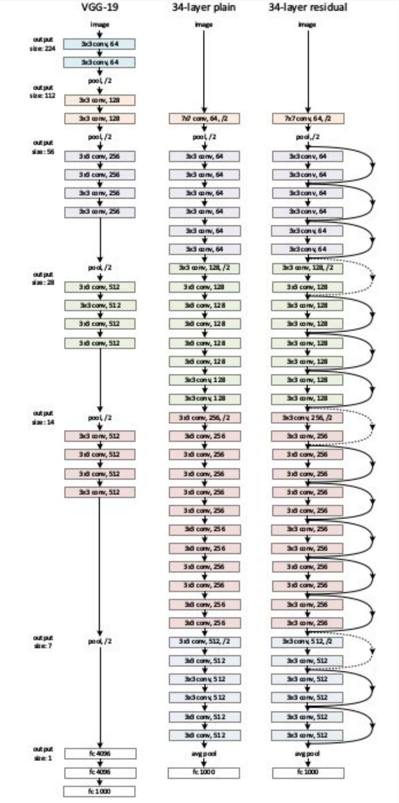


Fig. 1. GRNN block diagram.

1991

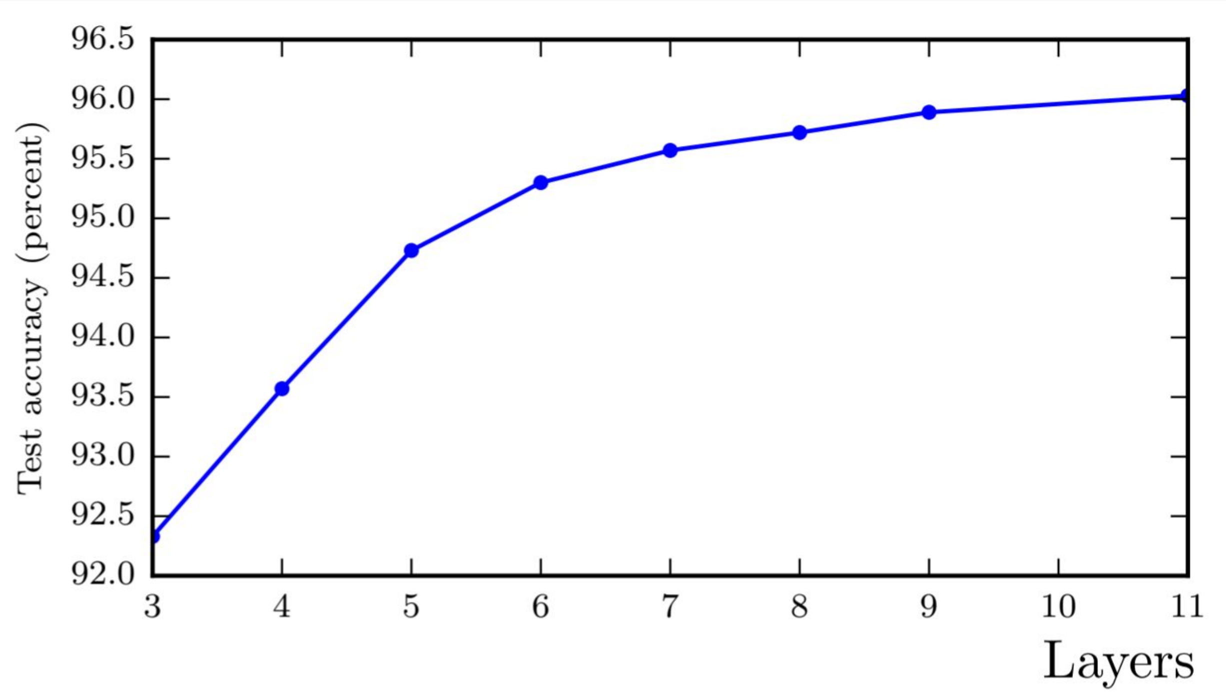
VS



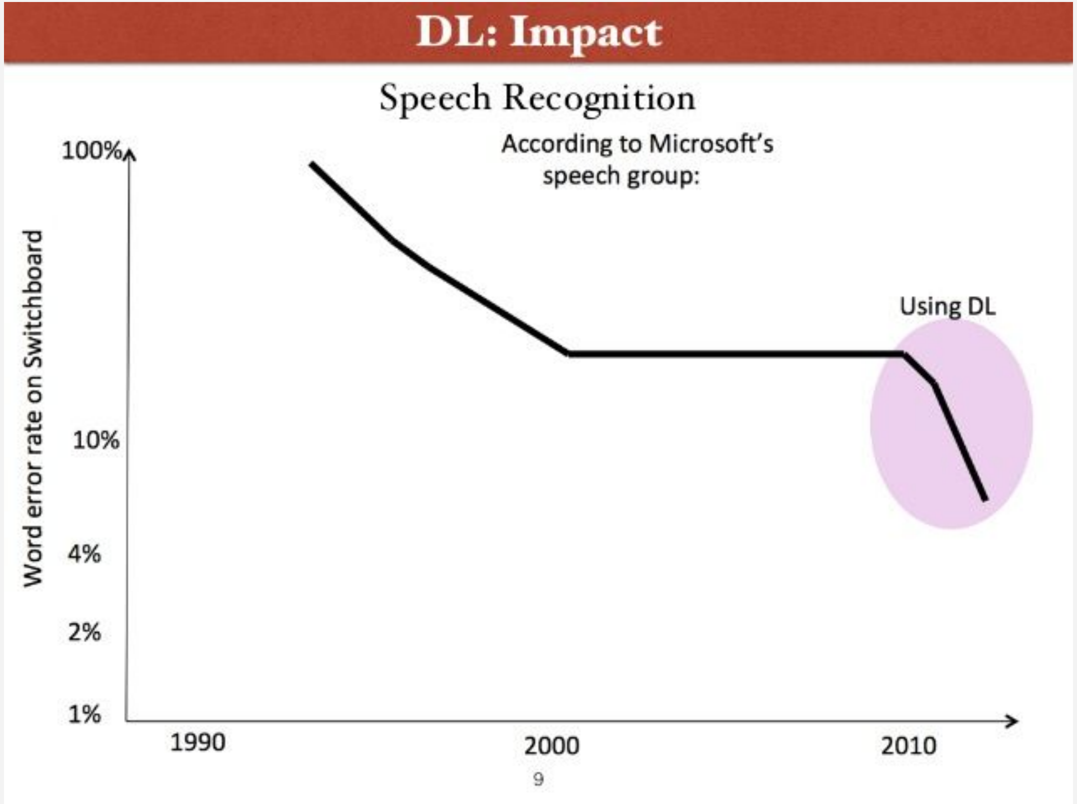
2016

Specht, Donald F. "A general regression neural network." IEEE transactions on neural networks 2.6 (1991): 568-576.

The Benefit of Going Deeper

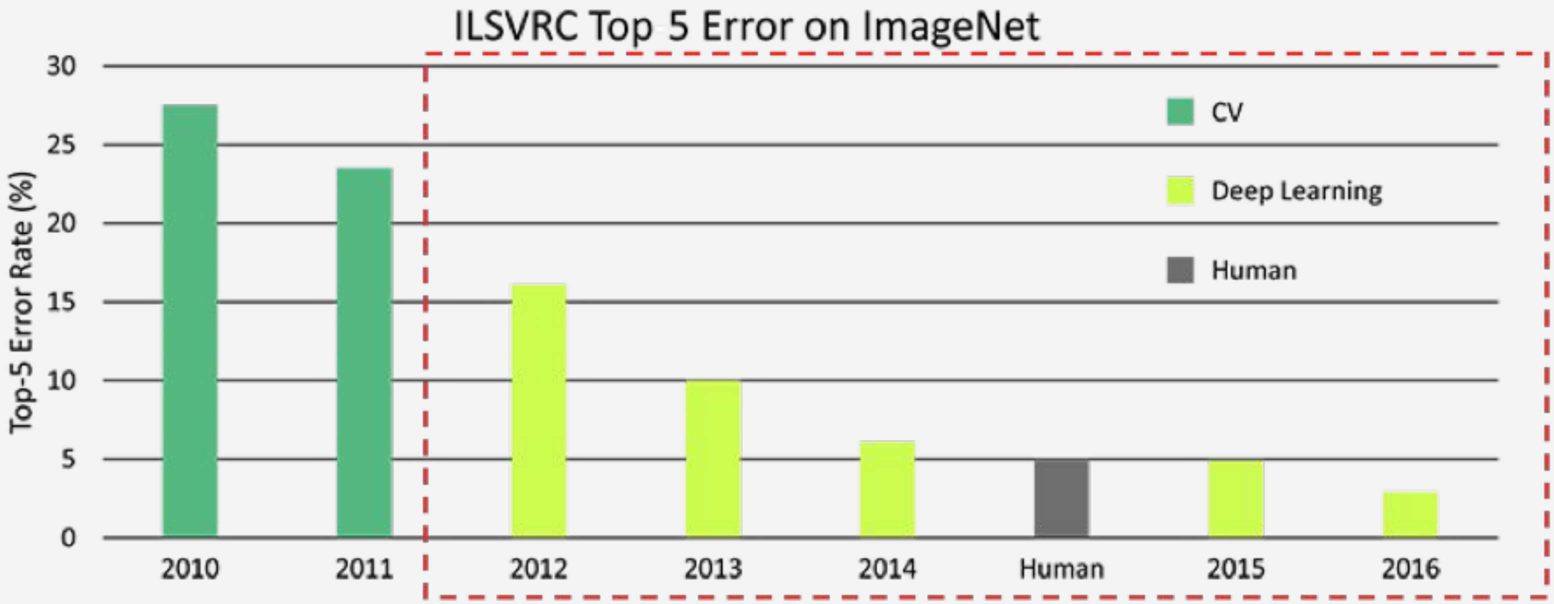


The Dominance of DL



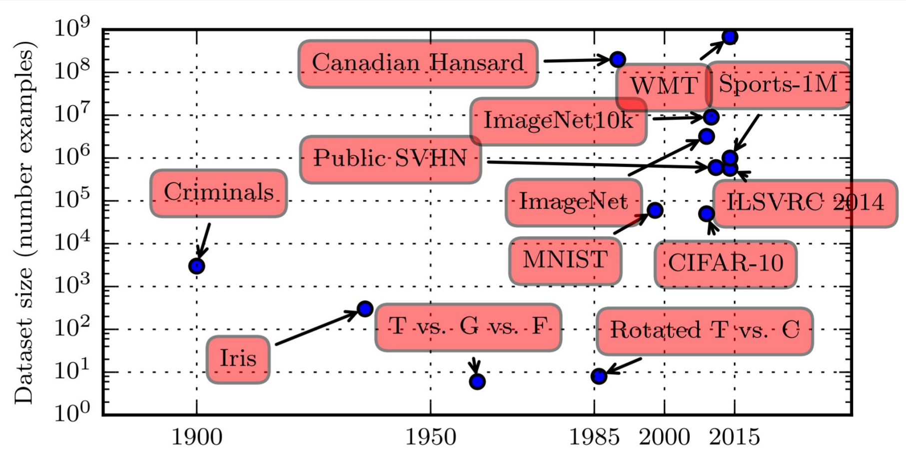
https://www.slideshare.net/roelofp/220115dlmeetup

The Dominance of DL



The introduction of Deep Learning techniques drove performance on image categorization from 30% error rates in 2010, down to <2% in 2017

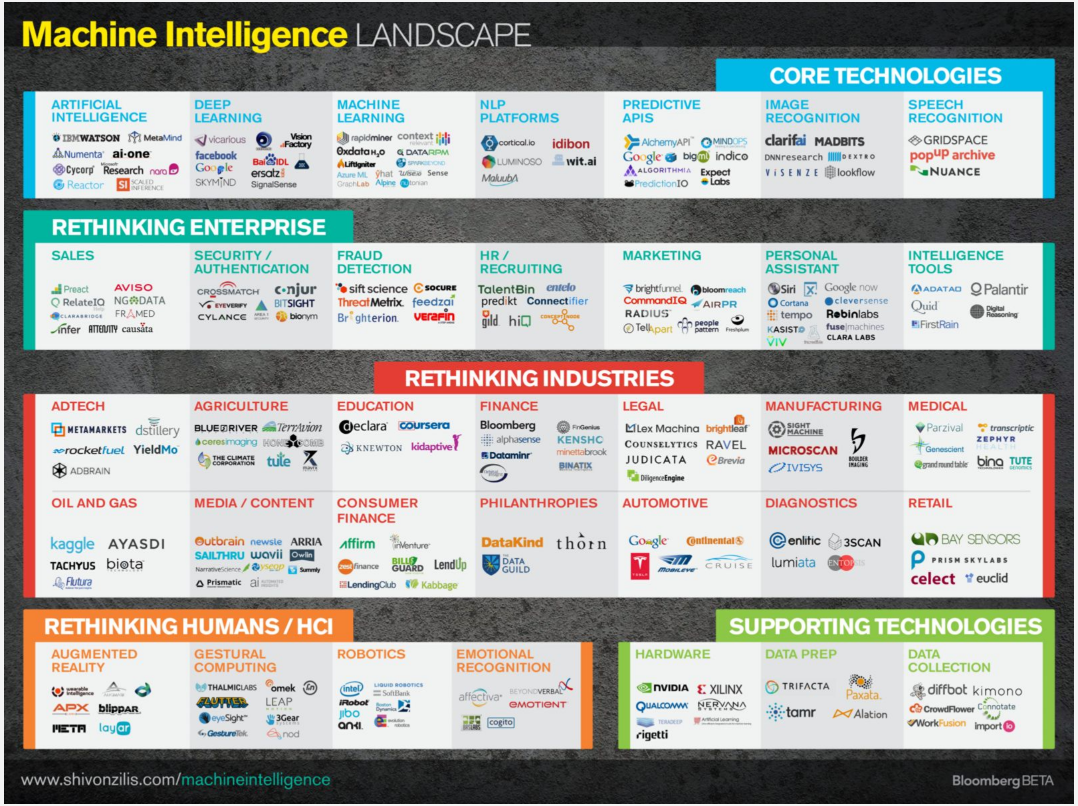
Main Reasons Behind Deep Learning's Success



Data

Hardware

Deep Learning Landscape



https://medium.com/@shivon/the-current-state-of-machine-intelligence-f76c20db2fe1

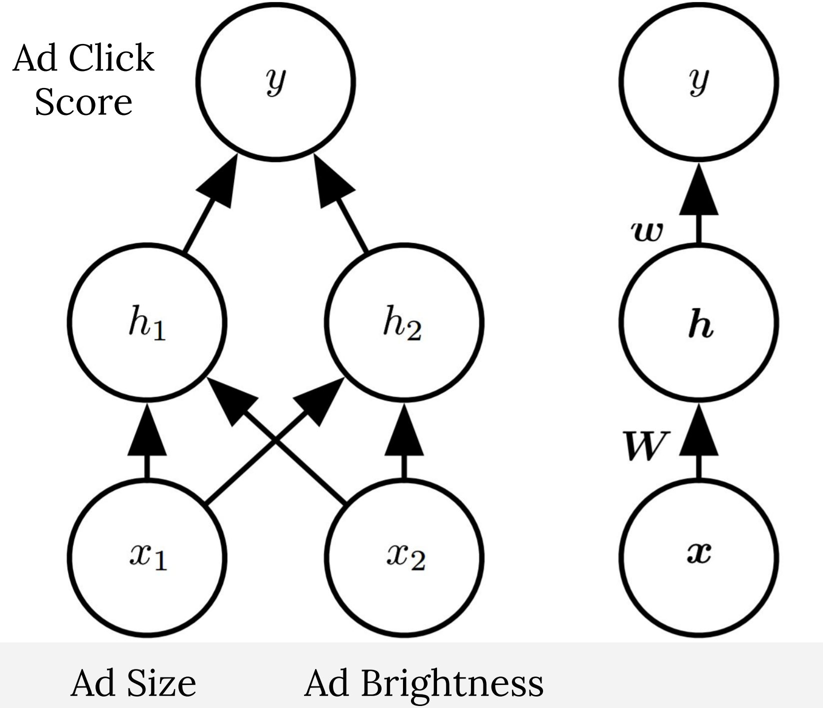
Today

- Overview
- **Basic Feedforward Networks and Core Concepts**
  - Optimization
  - Regularization
- Convolutional Networks
- Recurrent Networks
- Generative Models
- Research Frontiers

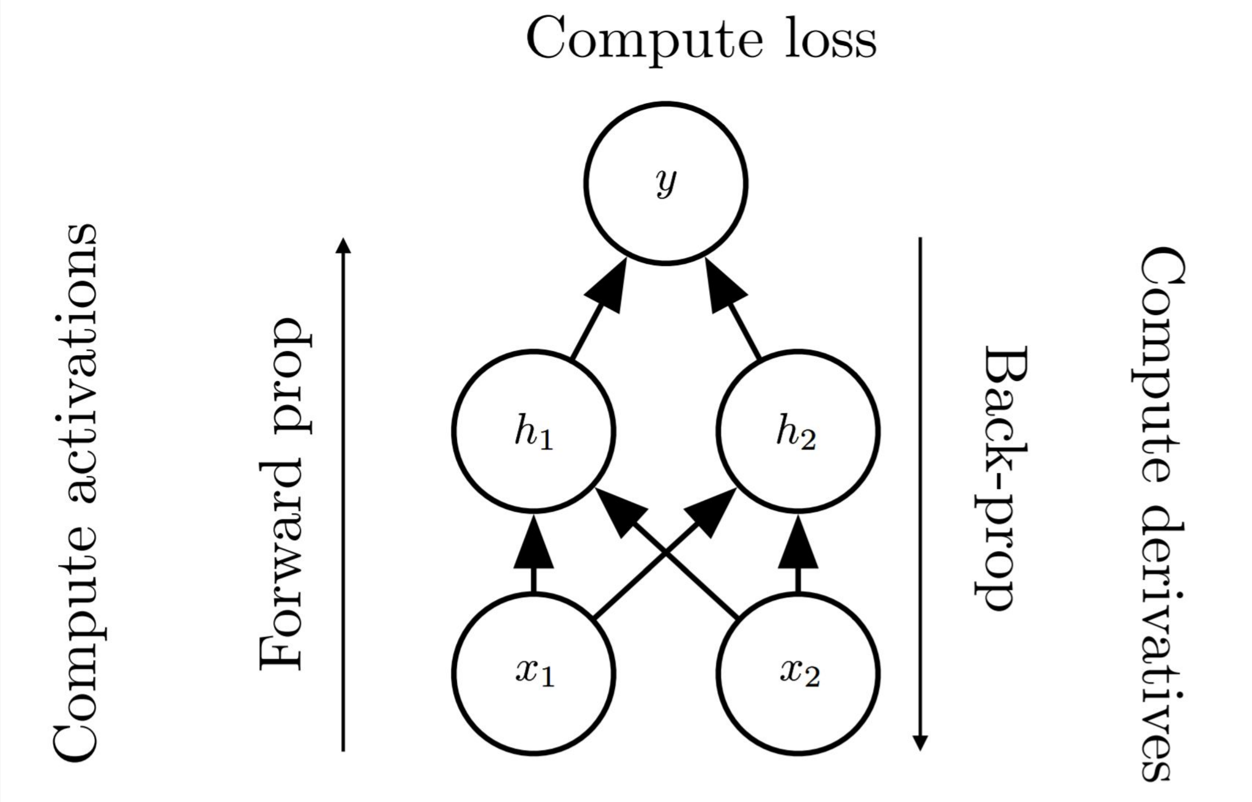
Slides adapted from Ian Goodfellow.

Basic Feedforward Networks

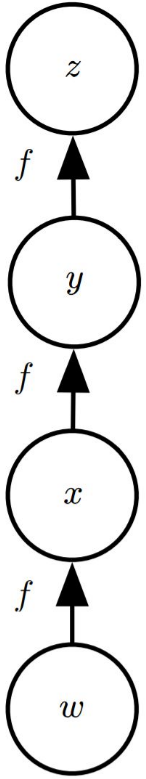
$$f(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c}, \boldsymbol{w}, b) = \boldsymbol{w}^\top \max\{0, \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{c}\} + b.$$



Backpropagation



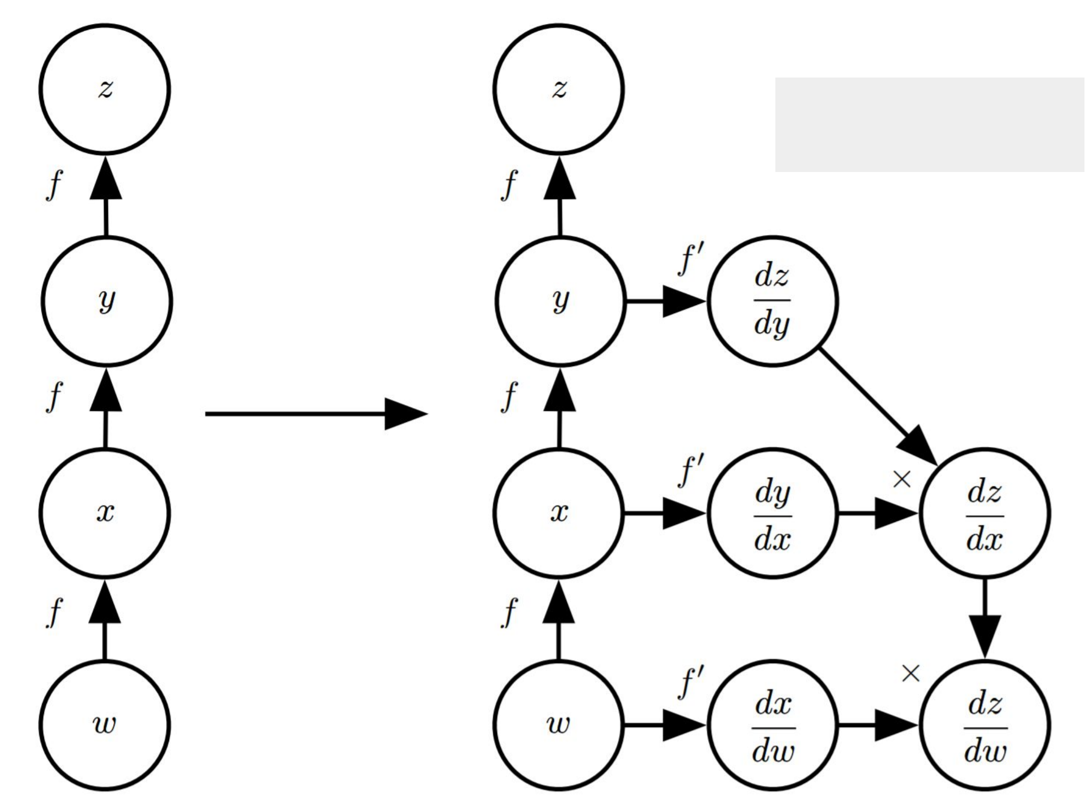
Backpropagation



$$
\begin{aligned}
&\frac{\partial z}{\partial w} \\
=&\frac{\partial z}{\partial y}\frac{\partial y}{\partial x}\frac{\partial x}{\partial w} \\
=&f'(y)f'(x)f'(w) \\
=&f'(f(f(w)))f'(f(w))f'(w)
\end{aligned}
$$

Back-prop avoids computing this twice

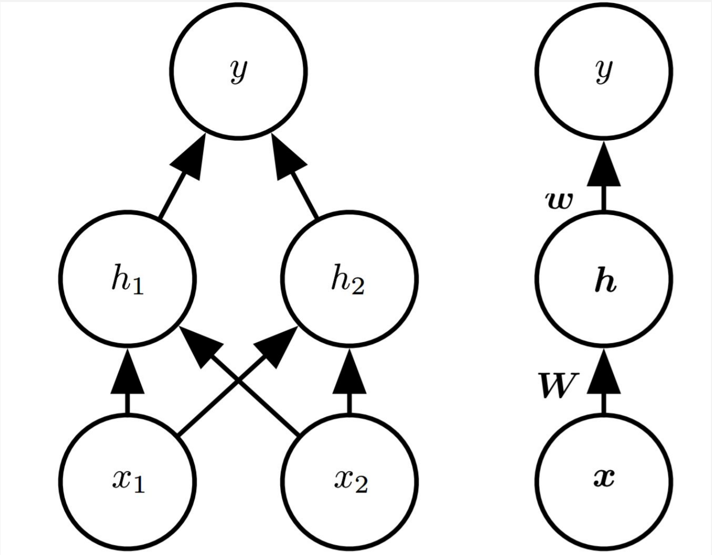
Backpropagation - How It Is Done



Backpropagation - PyTorch Example

```
import torch
from torch.autograd import Variable

x = Variable(torch.ones(2), requires_grad=True)

W = Variable(torch.ones(2), requires_grad=True)
c = Variable(torch.ones(2), requires_grad=True)

w = Variable(torch.ones(2,1), requires_grad=True)
b = Variable(torch.ones(1), requires_grad=True)

h = torch.relu(x*W + c)
y = torch.matmul(w.t(),h) + b

y_target = Variable(torch.zeros(1))
loss  = (y - y_target)**2

loss.backward()
```

$$f(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c}, \boldsymbol{w}, b) = \boldsymbol{w}^\top \max\{0, \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{c}\} + b.$$



Optimization

Why backpropagation is useful?

It helps us to calculate the gradient of a model’s parameters, so we can use gradient based optimization techniques.

Recap from last lecture:

$$\text{minimize}_\theta \ e \doteq \mathbb{E}_{(x,y)\sim D}[L(y, f(x;\theta))]$$

Find the best θ that minimizing the expected loss.

Common Optimization Algorithms in DL

Stochastic Gradient Descent (SGD)

Instead of calculating the gradient over the whole training set, only do it over a few examples (called a minibatch).

No need to fit the whole data into memory; Enables online learning.

Adam

A more advanced SGD extension. Works well with noisy gradient and large data problems.

Optimization - PyTorch Example

```
optimizer = optim.SGD(model.parameters(), lr = 0.01, momentum=0.9)

or

optimizer = optim.Adam([var1, var2], lr = 0.0001)


for input, target in dataset:

    optimizer.zero_grad()

    output = model(input)
    loss = loss_fn(output, target)
    loss.backward()

    optimizer.step()
```

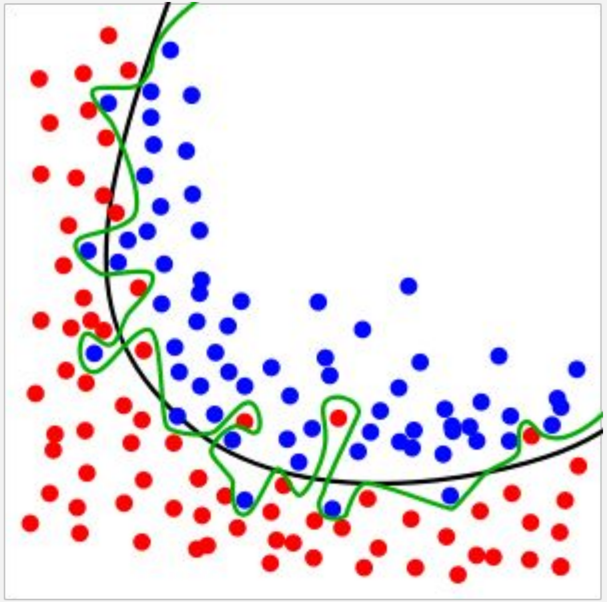
Regularization

L1 Norm

Prefer sparse weights

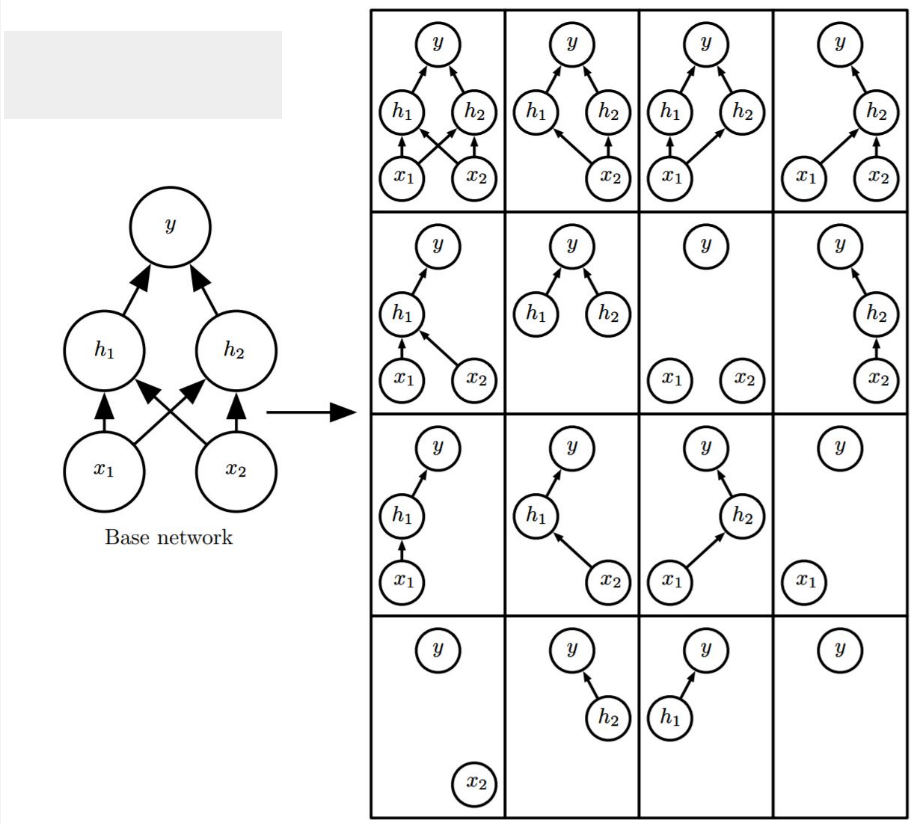
L2 Norm

Prefer smaller weights



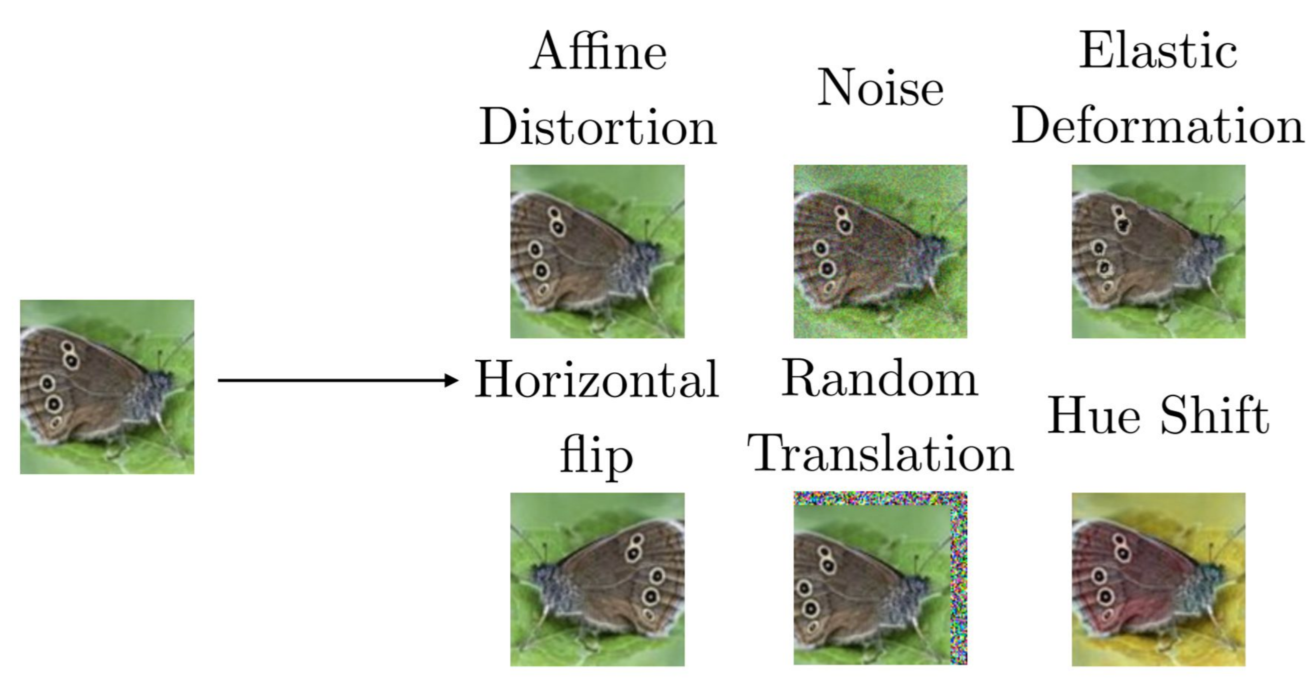
Regularization

Dropout



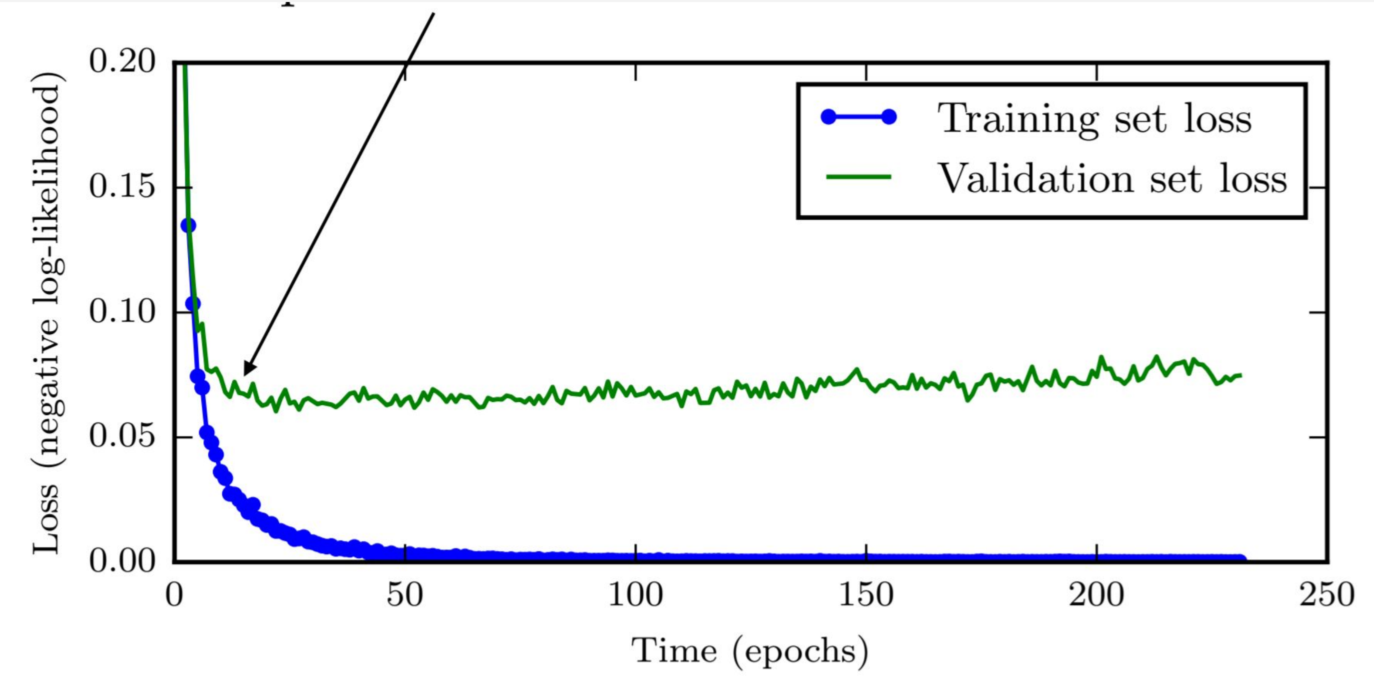
Regularization

Data Augmentation



Regularization

Early Stopping



Today

- Overview
- Basic Feedforward Networks and Core Concepts
    - Optimization
    - Regularization
- **Convolutional Networks**
- Recurrent Networks
- Generative Models
- Research Frontiers

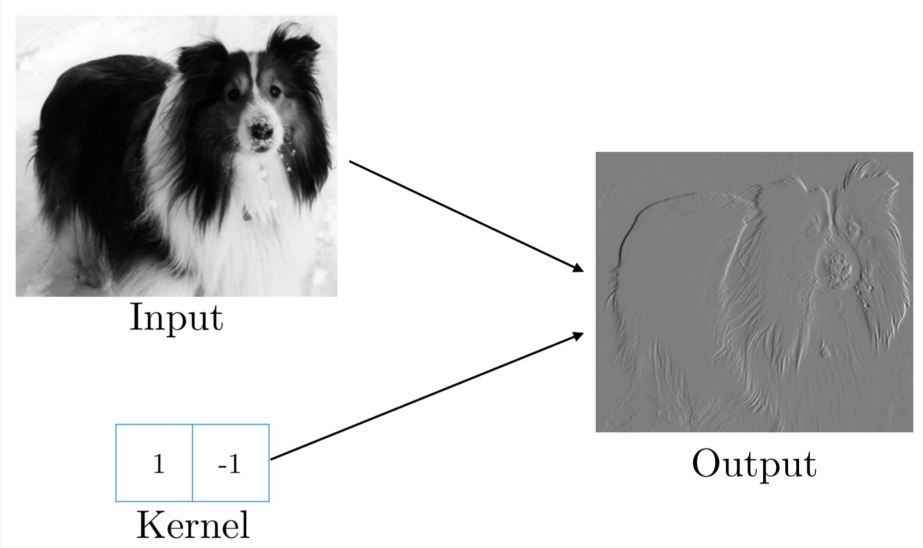
Data and Neural Network Models

Static Data	Dynamic Data	Unsupervised Data
Convolutional Neural Networks	Recurrent Neural Networks	Generative Neural Networks

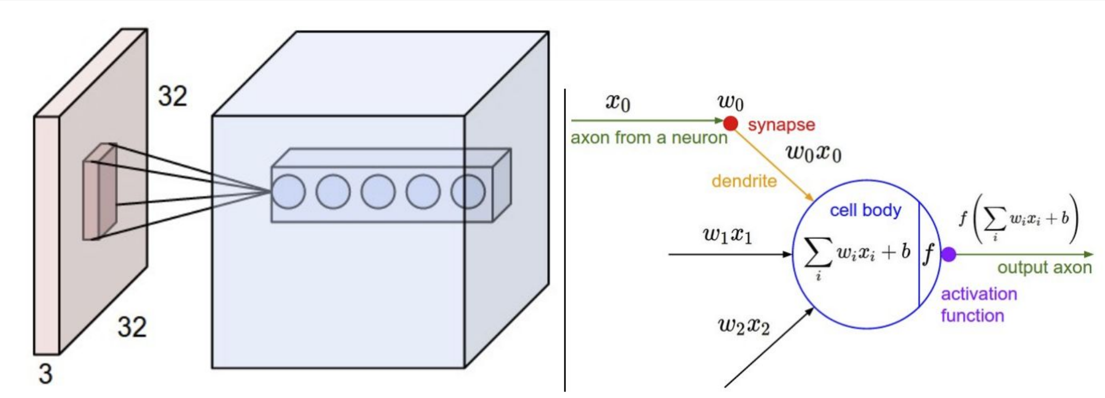
Convolutional Networks

Convolution

A local operation that extracts information from data.



Convolutional Networks

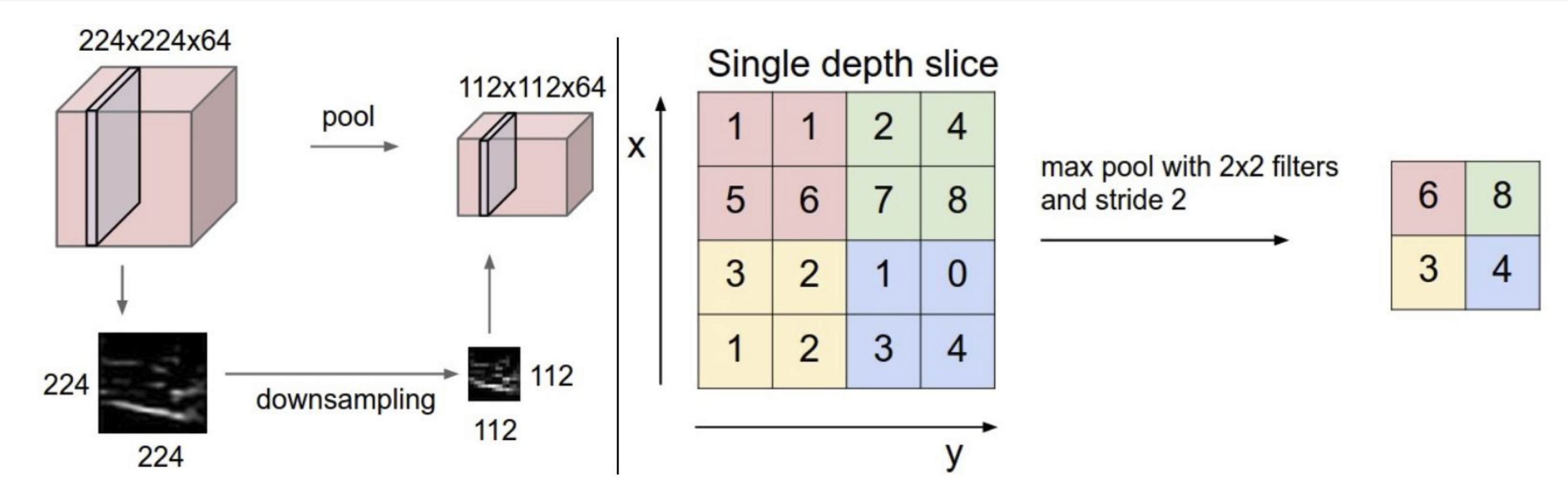


Convolution in action:

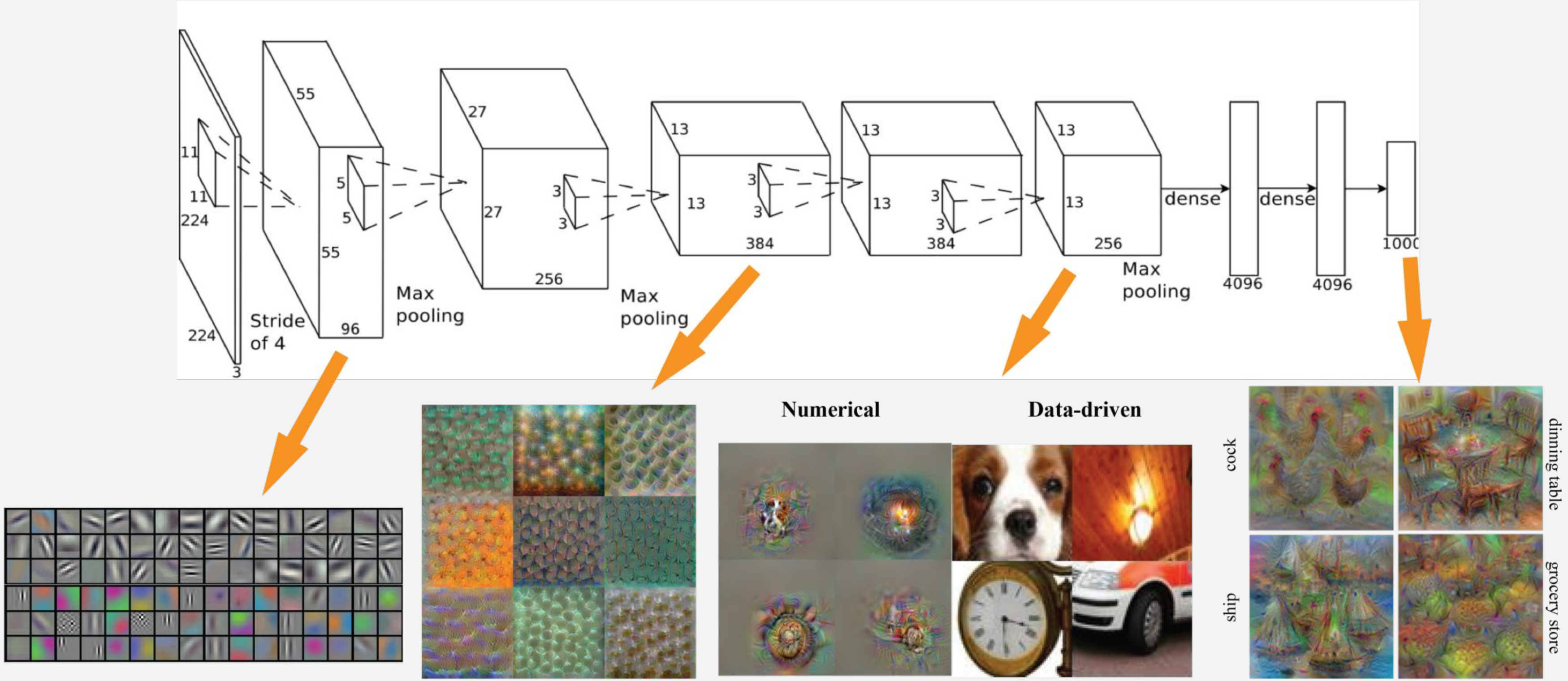
http://cs231n.github.io/convolutional-networks/

Convolutional Networks

Pooling



Convolutional Networks



Conv 1: Edge+Blob

Conv 3: Texture

Conv 5: Object Parts

Fc8: Object Classes

Convolutional Networks

Good for:

Data with translation invariance and shared statistics.

Data that can benefitted from different levels of abstraction.

Not so good for:

Dynamic data.

Today

- Overview
- Basic Feedforward Networks and Core Concepts
  - Optimization
  - Regularization
- Convolutional Networks
- **Recurrent Networks**
- Generative Models
- Research Frontiers

Data and Neural Network Models

Static Data	Dynamic Data	Unsupervised Data
Convolutional Neural Networks	Recurrent Neural Networks	Generative Neural Networks

Recurrent Networks

Dynamic Data

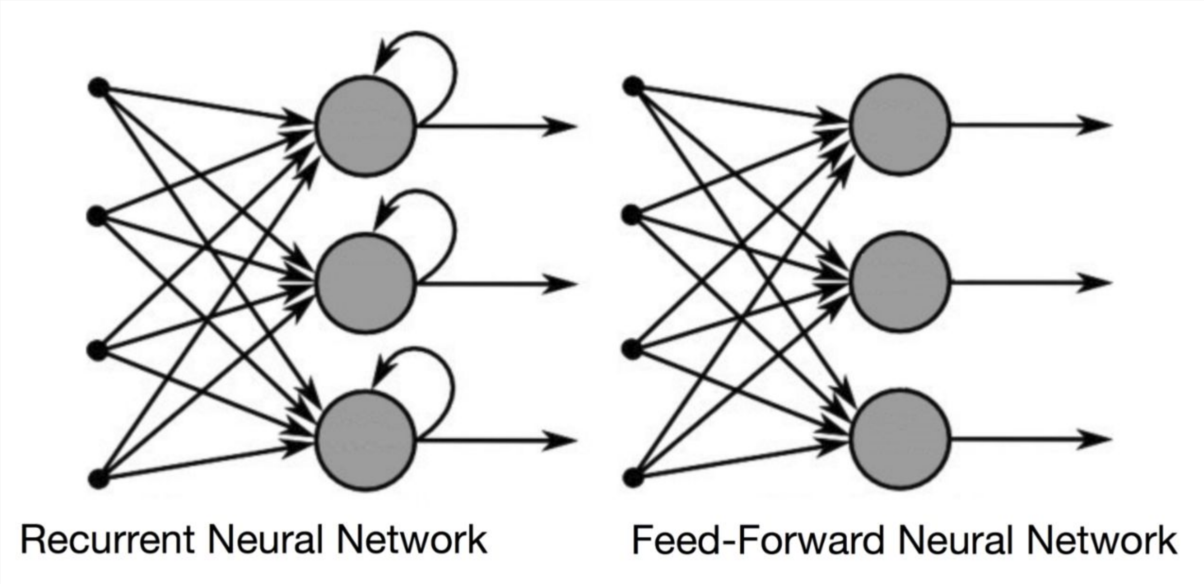
Data changes over time.

Language

Video

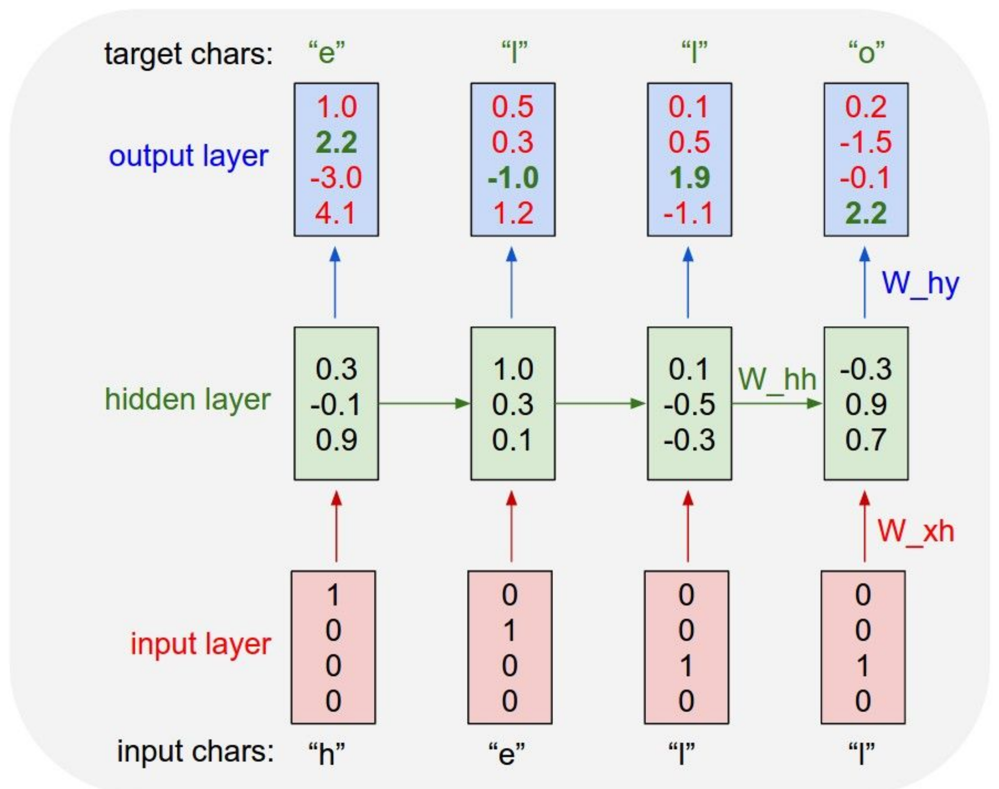
Recurrent Networks

A model's output is not just depending on the current input



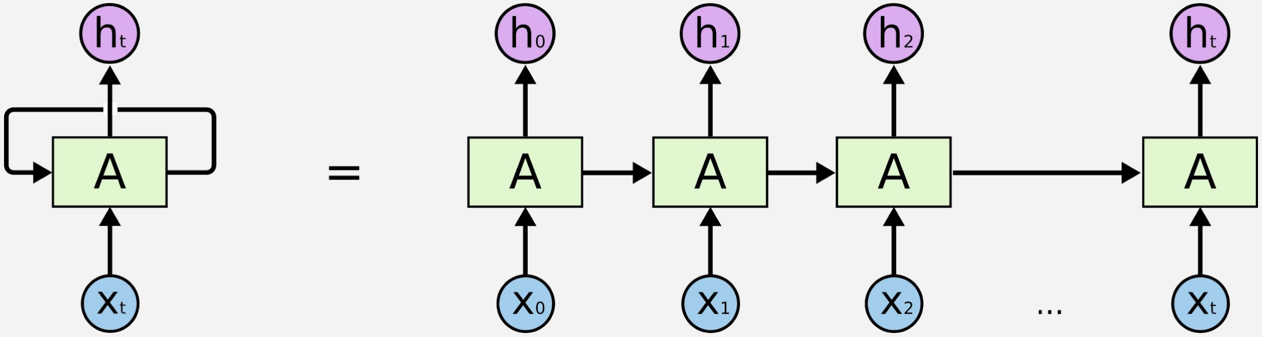
https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5

Recurrent Networks

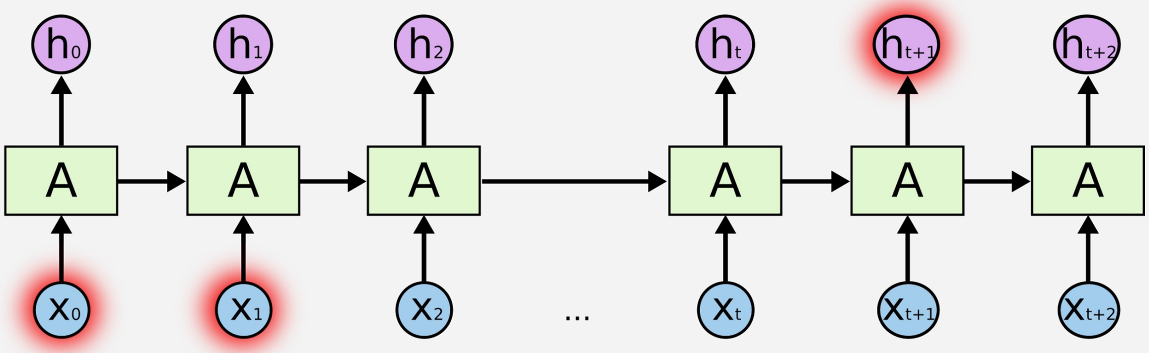


http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Unrolled RNNs



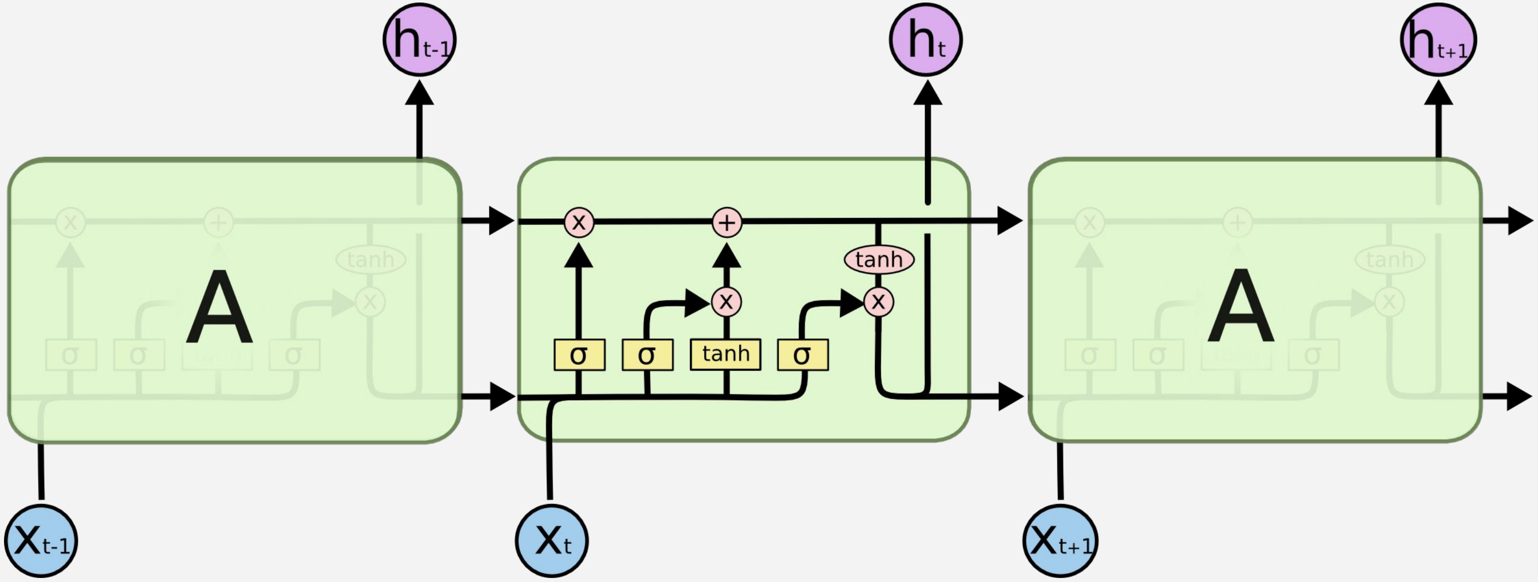
Long term relations



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short-Term Memory (LSTM) Networks

Being able to remember… and forget!



Recurrent Networks

Good for:

Dynamic data.

Not so good:

Might be tricky to train.

An interesting read.

Today

- Overview
- Basic Feedforward Networks and Core Concepts
  - Optimization
  - Regularization
- Convolutional Networks
- Recurrent Networks
- **Generative Models**
- Research Frontiers

Data and Neural Network Models

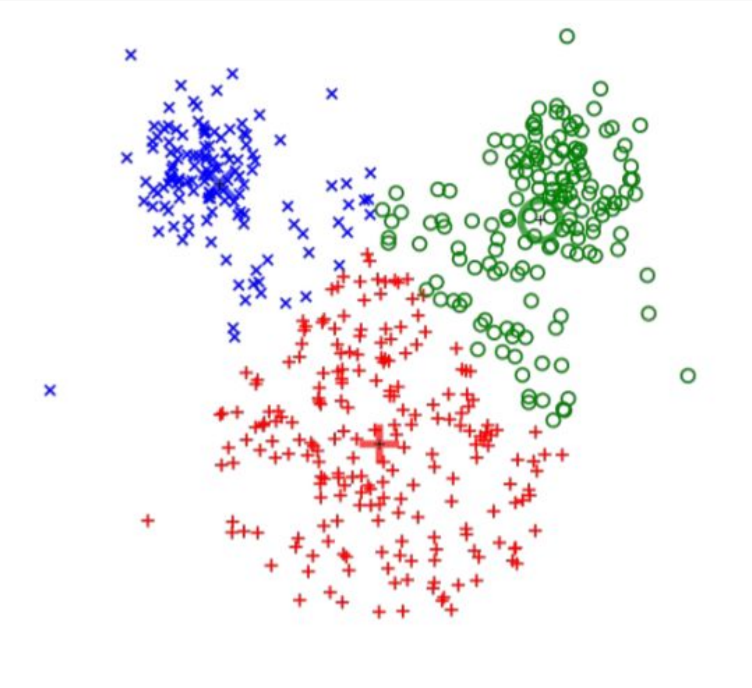
Static Data	Dynamic Data	Unsupervised Data
Convolutional Neural Networks	Recurrent Neural Networks	Generative Neural Networks

Generative Models

We have data, but no labels.

Goal

Recover underlying structures of the data.

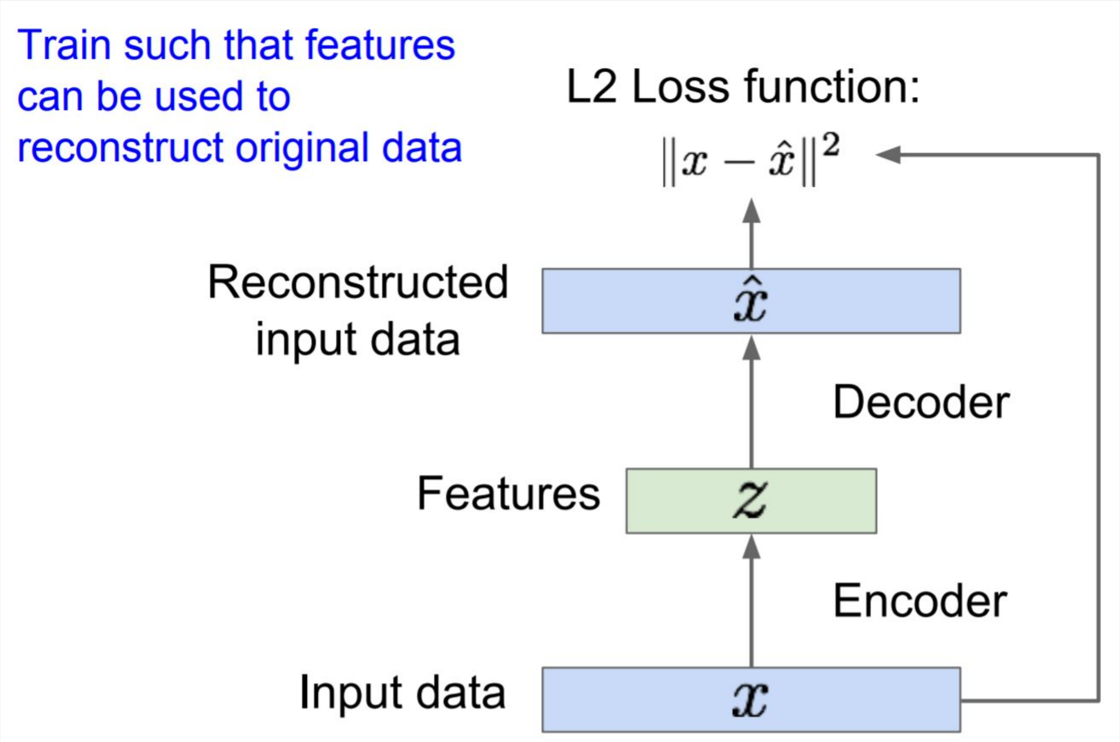


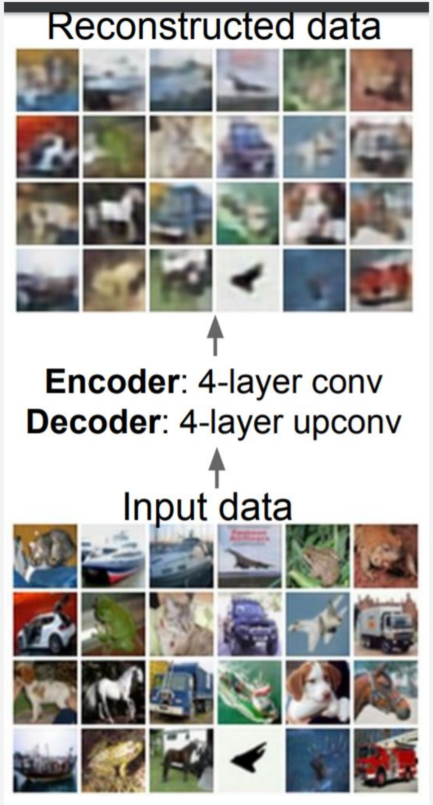
Generative Models

- Realistic samples for artwork, super-resolution, colorization, etc.

http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf

Autoencoder

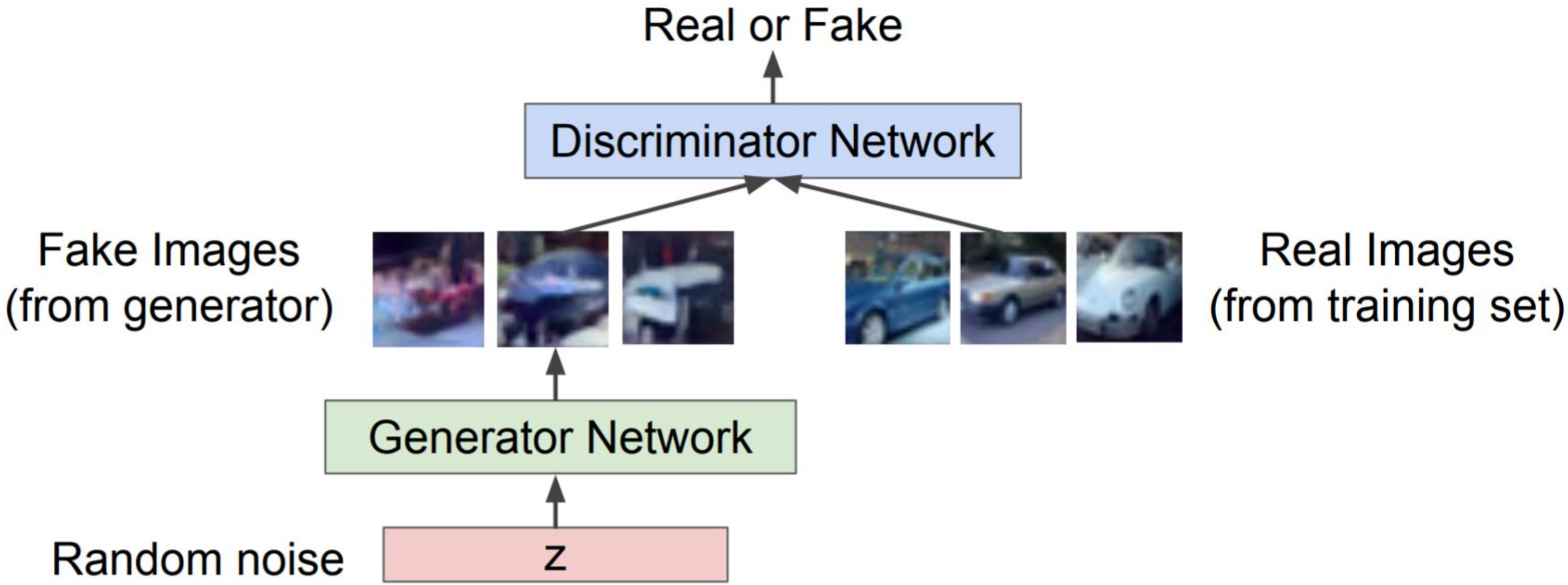




http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf

Generative Adversarial Networks

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images



http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf

Today

- Overview
- Basic Feedforward Networks and Core Concepts
  - Optimization
  - Regularization
- Convolutional Networks
- Recurrent Networks
- Generative Models
- **Research Frontiers**

Research Frontiers

Deeper Networks

Vanishing gradient, instability -> ResNet

More Efficient Networks

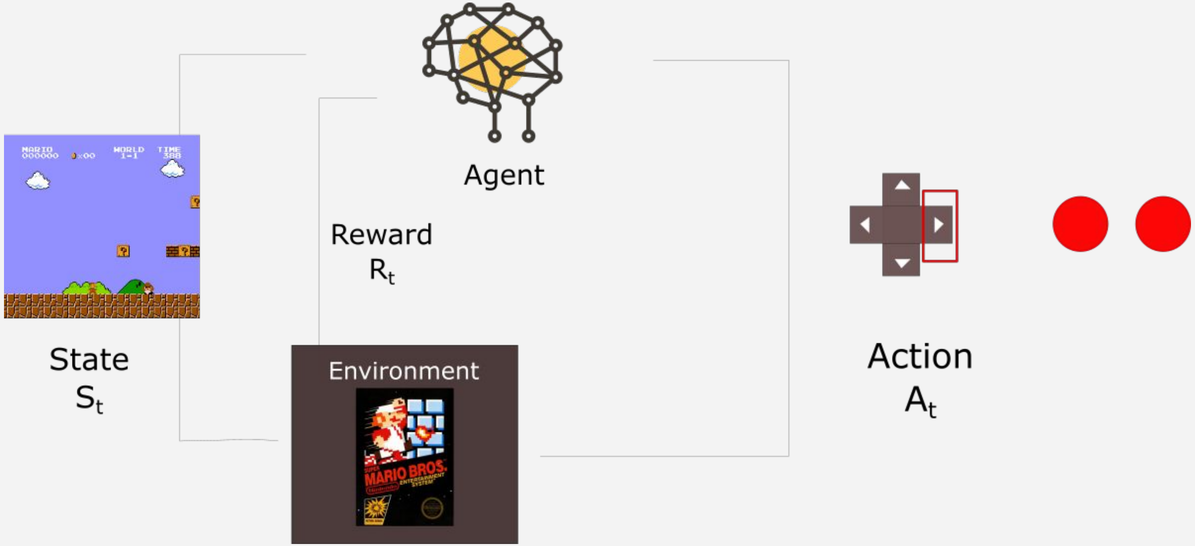
Network compression / Binary networks

Understanding Networks

The explainability of neural networks

Research Frontiers

Reinforcement Learning is trying to solve a very different problem than standard ML: instead of supervision, we are given a vague signal called ‘reward’.



https://medium.freecodecamp.org/an-introduction-to-reinforcement-learning-4339519de419

Summary

- Overview
- Basic Feedforward Networks and Core Concepts
  - Optimization
  - Regularization
- Convolutional Networks
- Recurrent Networks
- Generative Models
- Research Frontiers

Further Readings:

Deep Learning by Ian Goodfellow and Yoshua Bengio and Aaron Courville link