Foundations of Artificial Intelligence

14. Deep Learning

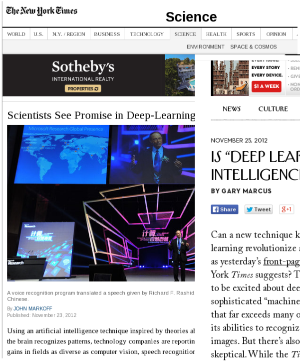
Learning from Raw Data

Joschka Boedecker and Wolfram Burgard and
Frank Hutter and Bernhard Nebel and Michael Tangermann

Albert-Ludwigs-Universität Freiburg

July 10, 2019

Motivation: Deep Learning in the News


The New York Times

Science

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION

ENVIRONMENT SPACE & COSMOS

Scientists See Promise in Deep-Learning

A voice recognition program translated a speech given by Richard F. Rashid Chinese.

By JOHN MARKOFF
Published: November 23, 2012

Using an artificial intelligence technique inspired by theories al the brain recognizes patterns, technology companies are reportin gains in fields as diverse as computer vision, speech recognitior


THE NEW YORKER

NEWS CULTURE BOOKS & FICTION SCIENCE & TECH BUSINESS HUMOR MAGAZINE ARCHIVE SUBSCRIBE

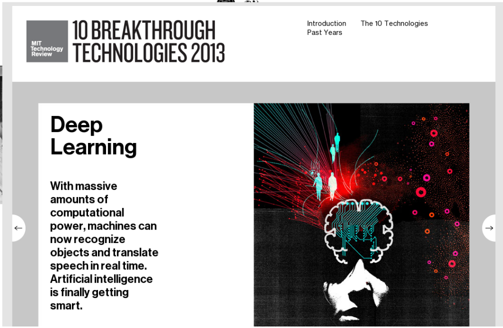
NOVEMBER 25, 2012

IS "DEEP LEARNING" A REVOLUTION IN ARTIFICIAL INTELLIGENCE?

BY GARY MARCUS

Share Tweet +1

Can a new technique known as deep learning revolutionize artificial intelligence, as yesterday's front-page article at the New York *Times* suggests? There is good reason to be excited about deep learning, a sophisticated "machine learning" algorithm that far exceeds many of its predecessors in its abilities to recognize syllables and images. But there's also good reason to be skeptical. While the *Times* reports that "advances in an artificial intelligence technology that can recognize patterns offer


MIT Technology Review

10 BREAKTHROUGH TECHNOLOGIES 2013

Introduction
Past Years
The 10 Technologies

Deep Learning

With massive amounts of computational power, machines can now recognize objects and translate speech in real time. Artificial intelligence is finally getting smart.

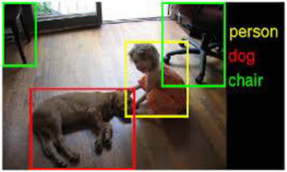
Lecture Overview

Lecture Overview

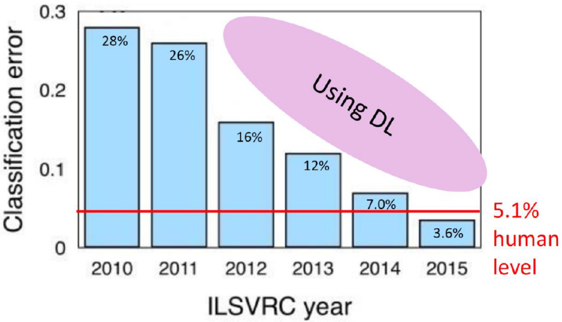
Motivation: Why is Deep Learning so Popular?

- Excellent empirical results, e.g., in computer vision

Object recognition



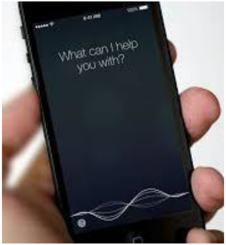
Self-driving cars



Motivation: Why is Deep Learning so Popular?

- Excellent empirical results, e.g., in speech recognition

Speech recognition



Auto-Translator

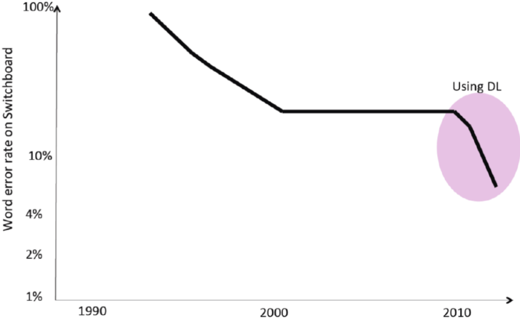
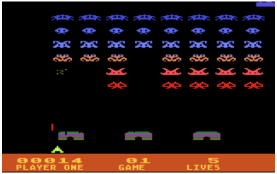


Image credit: Yoshua Bengio (data from Microsoft speech group)

Motivation: Why is Deep Learning so Popular?

- Excellent empirical results, e.g., in reasoning in games
  - Superhuman performance in playing Atari games [Mnih et al, Nature 2015]



  - Beating the world's best Go player [Silver et al, Nature 2016]

An Exciting Approach to AI: Learning as an Alternative to Traditional Programming

- We don't understand how the human brain solves certain problems
  - Face recognition
  - Speech recognition
  - Playing Atari games
  - Picking the next move in the game of Go
- We can nevertheless learn these tasks from data/experience

An Exciting Approach to AI: Learning as an Alternative to Traditional Programming

- We don't understand how the human brain solves certain problems
  - Face recognition
  - Speech recognition
  - Playing Atari games
  - Picking the next move in the game of Go
- We can nevertheless learn these tasks from data/experience
- If the task changes, we simply re-train

An Exciting Approach to AI: Learning as an Alternative to Traditional Programming

- We don't understand how the human brain solves certain problems
  - Face recognition
  - Speech recognition
  - Playing Atari games
  - Picking the next move in the game of Go
- We can nevertheless learn these tasks from data/experience
- If the task changes, we simply re-train
- We can construct computer systems that are too complex for us to understand anymore ourselves...
  - E.g., deep neural networks have millions of weights.
  - E.g., AlphaGo, the system that beat world champion Lee Sedol
    - David Silver, lead author of AlphaGo cannot say why a move is good
    - Paraphrased: "You would have to ask a Go expert."

An Exciting Approach to AI: Learning as an Alternative to Traditional Programming

- Learning from data / experience may be more human-like
  - Babies develop an intuitive understanding of physics in their first 2 years
  - Formal reasoning and logic comes much later in development

An Exciting Approach to AI: Learning as an Alternative to Traditional Programming

- Learning from data / experience may be more human-like
  - Babies develop an intuitive understanding of physics in their first 2 years
  - Formal reasoning and logic comes much later in development
- Learning enables fast reaction times
  - It might take a long time to train a neural network
  - But predicting with the network is very fast
  - Contrast this to running a planning algorithm every time you want to act

Lecture Overview

Some definitions

Representation learning

"a set of methods that allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification"

Some definitions

Representation learning

"a set of methods that allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification"

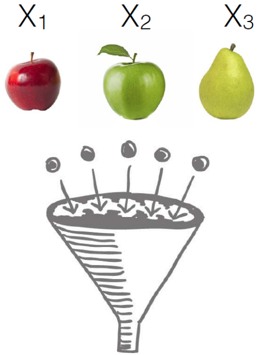
Deep learning

"representation learning methods with multiple levels of representation, obtained by composing simple but nonlinear modules that each transform the representation at one level into a [...] higher, slightly more abstract (one)"

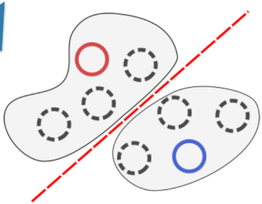
(LeCun et al., 2015)

Standard Machine Learning Pipeline

- Standard machine learning algorithms are based on high-level attributes or features of the data
- E.g., the binary attributes we used for decisions trees
- This requires (often substantial) feature engineering

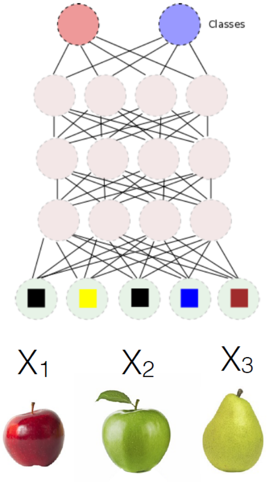


	Merkmale
X_1	rot, 3.5 cm
X_2	grün, 4 cm
X_3	grün, 10 cm

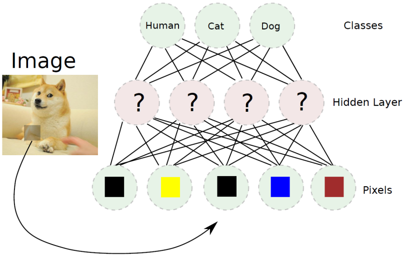


Representation Learning Pipeline

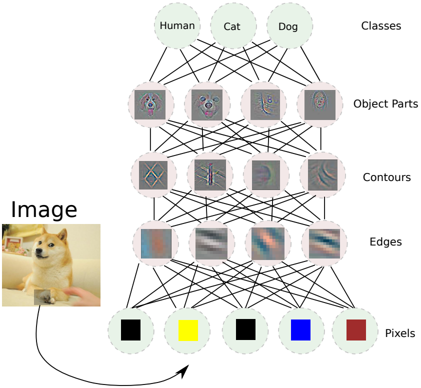
- Jointly learn features and classifier, directly from raw data
- This is also referrred to as end-to-end learning



Shallow vs. Deep Learning

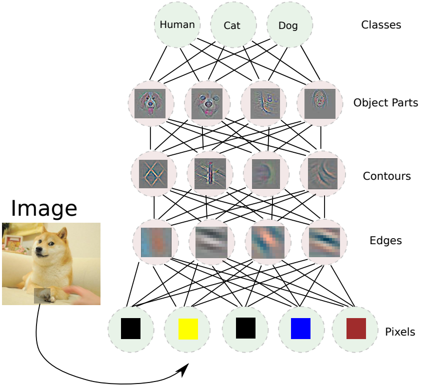


Shallow vs. Deep Learning



- Deep Learning: learning a hierarchy of representations that build on each other, from simple to complex

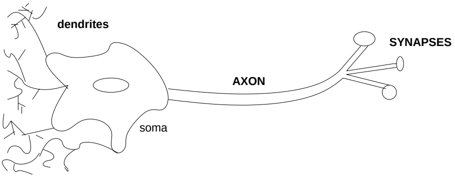
Shallow vs. Deep Learning



- Deep Learning: learning a hierarchy of representations that build on each other, from simple to complex
- Quintessential deep learning model: Multilayer Perceptrons

Biological Inspiration of Artificial Neural Networks

- Dendrites input information to the cell
- Neuron fires (has action potential) if a certain threshold for the voltage is exceeded
- Output of information by axon
- The axon is connected to dentrites of other cells via synapses
- Learning: adaptation of the synapse's efficiency, its synaptical weight



A Very Brief History of Neural Networks

- Neural networks have a long history
  - 1942: artificial neurons (McCulloch/Pitts)
  - 1958/1969: perceptron (Rosenblatt; Minsky/Papert)
  - 1986: multilayer perceptrons and backpropagation (Rumelhart)
  - 1989: convolutional neural networks (LeCun)

A Very Brief History of Neural Networks

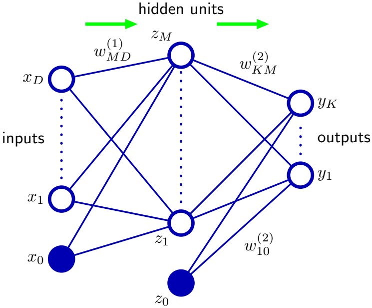
- Neural networks have a long history
  - 1942: artificial neurons (McCulloch/Pitts)
  - 1958/1969: perceptron (Rosenblatt; Minsky/Papert)
  - 1986: multilayer perceptrons and backpropagation (Rumelhart)
  - 1989: convolutional neural networks (LeCun)
- Alternative theoretically motivated methods outperformed NNs
  - Exaggerated expectations: “It works like the brain” (No, it does not!)

A Very Brief History of Neural Networks

- Neural networks have a long history
  - 1942: artificial neurons (McCulloch/Pitts)
  - 1958/1969: perceptron (Rosenblatt; Minsky/Papert)
  - 1986: multilayer perceptrons and backpropagation (Rumelhart)
  - 1989: convolutional neural networks (LeCun)
- Alternative theoretically motivated methods outperformed NNs
  - Exaggeraged expectations: "It works like the brain" (No, it does not!)
- Why the sudden success of neural networks in the last 5 years?
  - Data: Availability of massive amounts of labelled data
  - Compute power: Ability to train very large neural networks on GPUs
  - Methodological advances: many since first renewed popularization

Lecture Overview

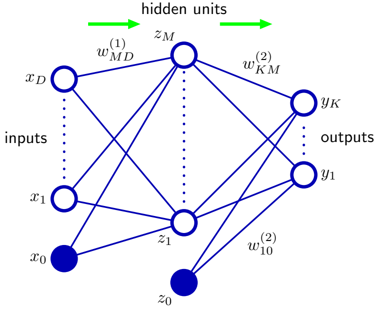
Multilayer Perceptrons



[figure from Bishop, Ch. 5]

- Network is organized in layers
  - Outputs of k-th layer serve as inputs of $k+1$th layer
- Each layer k only does quite simple computations:
  - Linear function of previous layer's outputs z_{k-1}: $a_k = W_k z_{k-1} + b_k$
  - Nonlinear transformation $z_k = h_k(a_k)$ through activation function h_k

Multilayer Perceptrons



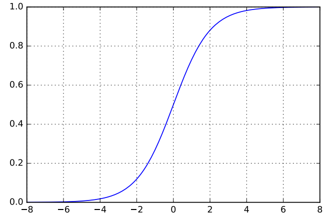
[figure from Bishop, Ch. 5]

- Network is organized in layers
  - Outputs of k-th layer serve as inputs of $k+1$th layer
- Each layer k only does quite simple computations:
  - Linear function of previous layer's outputs z_{k-1}: $a_k = W_k z_{k-1} + b_k$
  - Nonlinear transformation $z_k = h_k(a_k)$ through activation function h_k
- Parameters/weights w of the network: all W_k, b_k, flattened into a single vector

Activation Functions - Examples

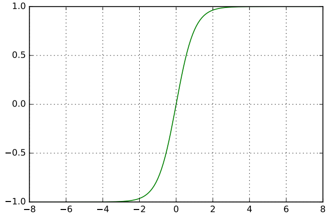
Logistic sigmoid activation function:

$$h_{logistic}(a) = \frac{1}{1 + \exp(-a)}$$



Logistic hyperbolic tangent activation function:

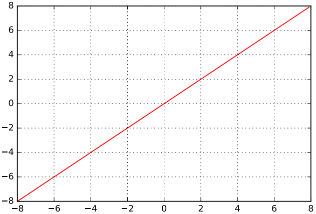
$$h_{tanh}(a) = \tanh(a)$$
$$= \frac{\exp(a) - \exp(-a)}{\exp(a) + \exp(-a)}\}$$



Activation Functions - Examples (cont.)

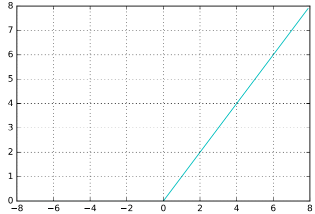
Linear activation function:

$$h_{linear}(a) = a$$



Rectified Linear (ReLU) activation function:

$$h_{relu}(a) = \max(0, a)$$



Output layer and loss functions

- For regression:
  - Single output neuron with linear activation function

$$\hat{y}(x, w) = h_{linear}(a) = a$$

  - Loss function: e.g., squared error:

$$L(w) = \frac{1}{2} \sum_{n=1}^{N} \{\hat{y}(x_n, w) - y_n\}^2$$

Output layer and loss functions

- For regression:
  - Single output neuron with linear activation function

$$\hat{y}(x, w) = h_{linear}(a) = a$$

  - Loss function: e.g., squared error:

$$L(w) = \frac{1}{2} \sum_{n=1}^{N} \{\hat{y}(x_n, w) - y_n\}^2$$

- For classification:
  - Single output unit with, e.g., logistic activation function:

$$\hat{y}(x, w) = h_{logistic}(a) = \frac{1}{1 + \exp(-a)}$$

  - Loss function: negative log likelihood of the data under the predictive distribution this specifies; (aka cross entropy):

$$L(w) = -\sum_{n=1}^{N} \{y_n \ln \hat{y}_n + (1 - y_n) \ln(1 - \hat{y}_n)\}$$

Optimizing a loss / error function

- Given training data $\mathcal{D} = \langle (x_i, y_i) \rangle_{i=1}^{N}$ and topology of an MLP
- Task: adapt weights w to minimize the loss:

$$\underset{w}{minimize}\ L(w; \mathcal{D})$$

Optimizing a loss / error function

- Given training data $\mathcal{D} = \langle (x_i, y_i) \rangle_{i=1}^{N}$ and topology of an MLP
- Task: adapt weights w to minimize the loss:

$$\underset{w}{minimize}\ L(w; \mathcal{D})$$

- We optimize this function by gradient-based optimization
  - We can compute gradients of $L(w; \mathcal{D})$
    - Efficiently, using a technique called backpropagation

Optimizing a loss / error function

- Given training data $\mathcal{D} = \langle (x_i, y_i) \rangle_{i=1}^{N}$ and topology of an MLP
- Task: adapt weights w to minimize the loss:

$$\underset{w}{minimize}\ L(w; \mathcal{D})$$

- We optimize this function by gradient-based optimization
  - We can compute gradients of $L(w; \mathcal{D})$
    - Efficiently, using a technique called backpropagation
  - Stochastic gradient descent (SGD)
    - We can use small batches of the data, i.e., $L(w; \mathcal{D}_{batch})$
    - This yields approximate gradients quickly

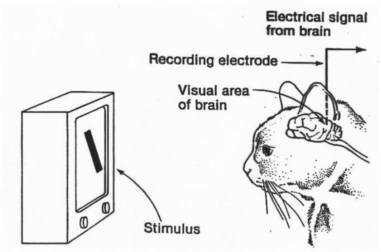
Lecture Overview

Lecture Overview

Historical context and inspiration from Neuroscience

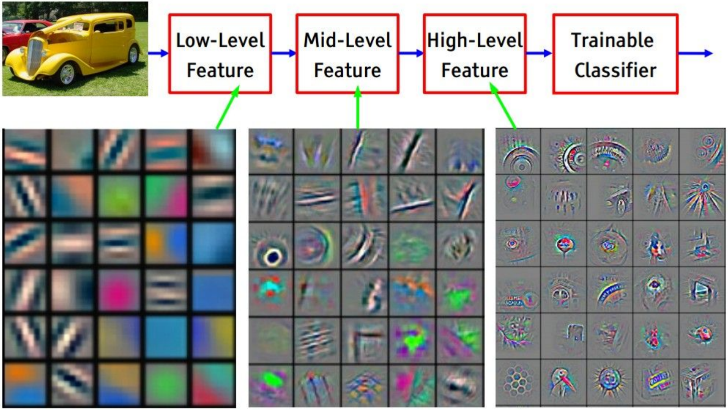
Hubel & Wiesel (Nobel prize 1981) found in several studies in the 1950s and 1960s:

- Visual cortex has feature detectors (e.g., cells with preference for edges with specific orientation)
  - edge location did not matter
- Simple cells as local feature detectors
- Complex cells pool responses of simple cells
- There is a feature hierarchy



Learned feature hierarchy

[From recent Yann LeCun slides]



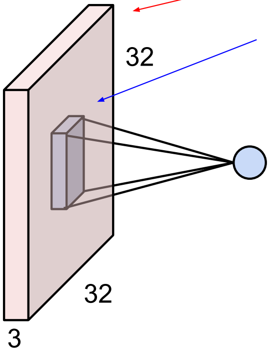
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

[slide credit: Andrej Karpathy]

Convolutions illustrated

32x32x3 image

5x5x3 filter w

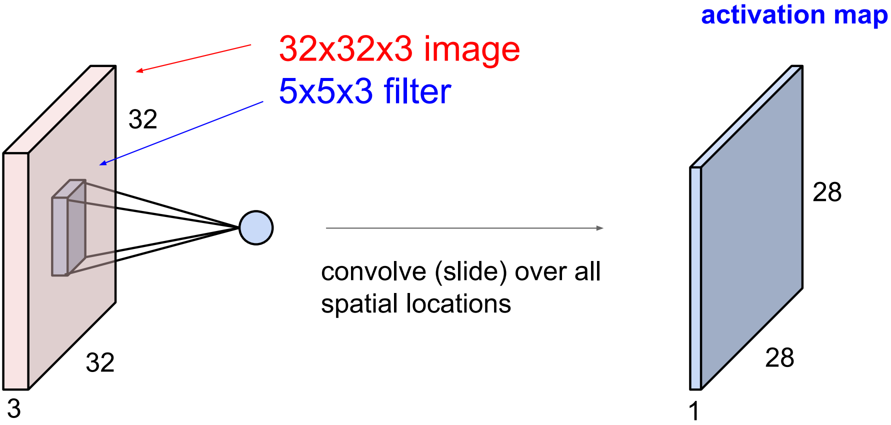


1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image (i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

[slide credit: Andrej Karpathy]

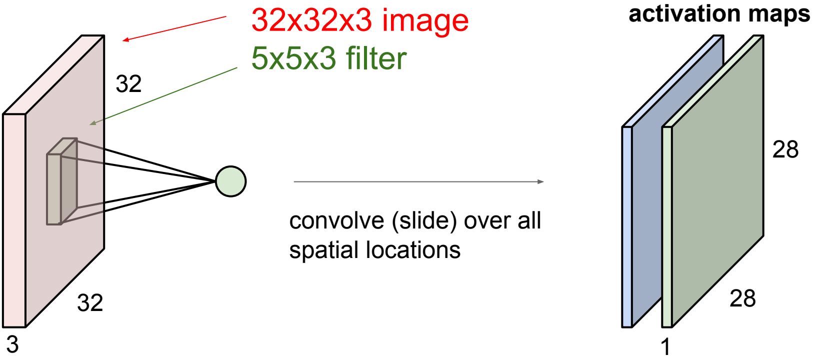
Convolutions illustrated (cont.)



[slide credit: Andrej Karpathy]

Convolutions – several filters

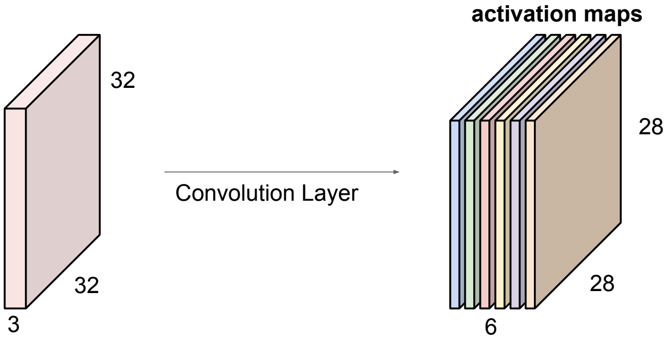
consider a second, green filter



[slide credit: Andrej Karpathy]

Convolutions – several filters

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:

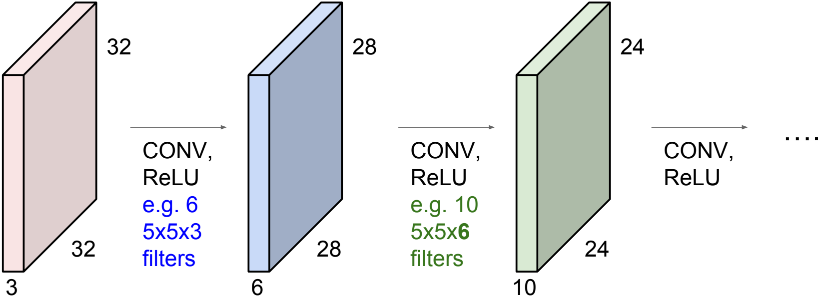


We stack these up to get a "new image" of size 28x28x6!

[slide credit: Andrej Karpathy]

Stacking several convolutional layers

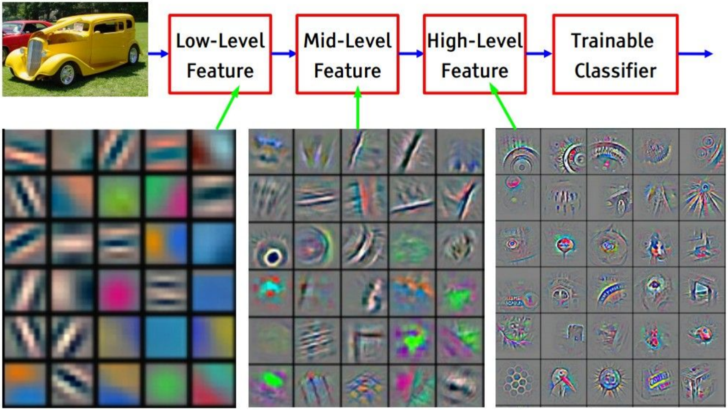
Convolutional layers stacked in a ConvNet



[slide credit: Andrej Karpathy]

Learned feature hierarchy

[From recent Yann LeCun slides]

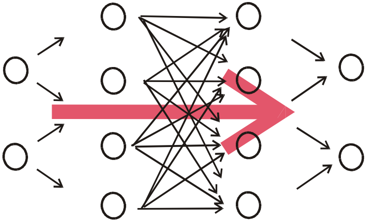


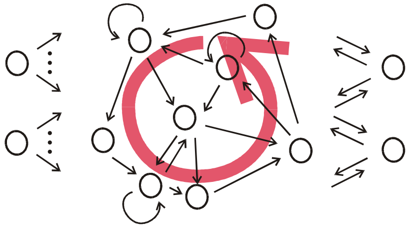
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

[slide credit: Andrej Karpathy]

Lecture Overview

Feedforward vs Recurrent Neural Networks



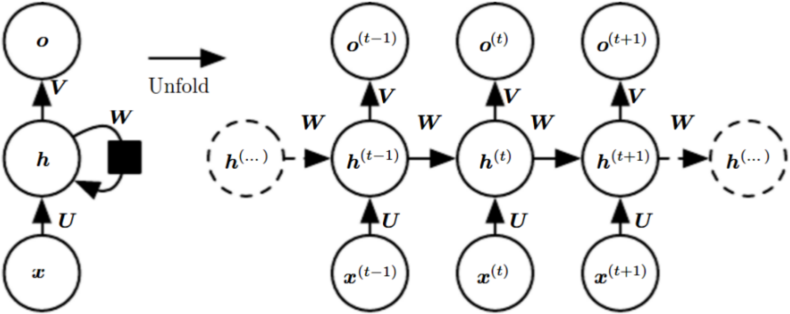


[Source: Jaeger, 2001]

Recurrent Neural Networks (RNNs)

- Neural Networks that allow for cycles in the connectivity graph
- Cycles let information persist in the network for some time (state), and provide a time-context or (fading) memory
- Very powerful for processing sequences
- Implement dynamical systems rather than function mappings, and can approximate any dynamical system with arbitrary precision
- They are Turing-complete [Siegelmann and Sontag, 1991]

With fully connected hidden layer:



[Goodfellow et al'2016]

Sequence to sequence mapping

one to many

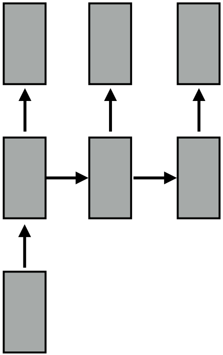
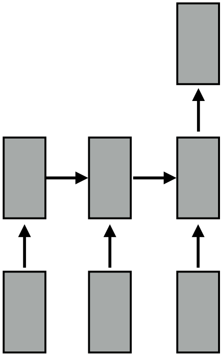


image caption generation

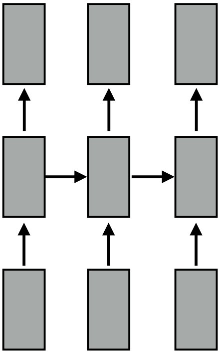
many to one



temporal classification

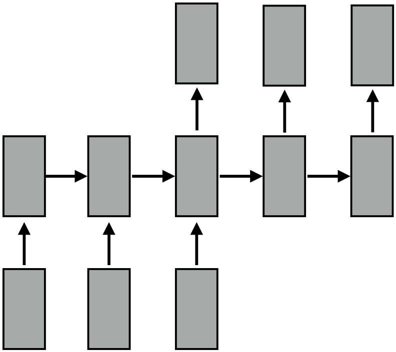
Sequence to sequence mapping (cont.)

many to many



video
frame labeling

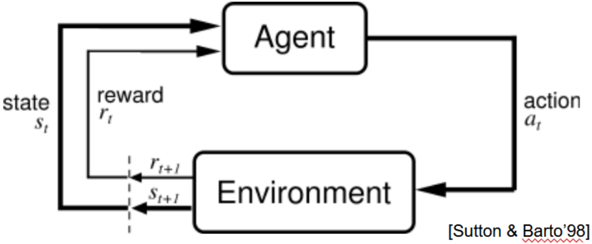
many to many



automatic
translation

Lecture Overview

Reinforcement Learning



- Finding optimal policies for MDPs
- Reminder: states $s \in S$, actions $a \in A$, transition model T, rewards r
- Policy: complete mapping $\pi : S \to A$ that specifies for each state s which action $\pi(s)$ to take

Deep Reinforcement Learning

- Policy-based deep RL
  - Represent policy $\pi : S \rightarrow A$ as a deep neural network with weights w
  - Evaluate w by "rolling out" the policy defined by w
  - Optimize weights to obtain higher rewards (using approx. gradients)
  - Examples: AlphaGo & modern Atari agents

Deep Reinforcement Learning

- Policy-based deep RL
  - Represent policy $\pi : S \rightarrow A$ as a deep neural network with weights w
  - Evaluate w by "rolling out" the policy defined by w
  - Optimize weights to obtain higher rewards (using approx. gradients)
  - Examples: AlphaGo & modern Atari agents
- Value-based deep RL
  - Basically value iteration, but using a deep neural network ($=$ function approximator) to generalize across many states and actions
  - Approximate optimal state-value function $U(s)$ or state-action value function $Q(s, a)$

Deep Reinforcement Learning

- Policy-based deep RL
  - Represent policy $\pi : S \rightarrow A$ as a deep neural network with weights w
  - Evaluate w by "rolling out" the policy defined by w
  - Optimize weights to obtain higher rewards (using approx. gradients)
  - Examples: AlphaGo & modern Atari agents
- Value-based deep RL
  - Basically value iteration, but using a deep neural network (= function approximator) to generalize across many states and actions
  - Approximate optimal state-value function $U(s)$ or state-action value function $Q(s, a)$
- Model-based deep RL
  - If transition model T is not known
  - Approximate T with a deep neural network (learned from data)
  - Plan using this approximate transition model

Deep Reinforcement Learning

- Policy-based deep RL
  - Represent policy $\pi : S \to A$ as a deep neural network with weights w
  - Evaluate w by "rolling out" the policy defined by w
  - Optimize weights to obtain higher rewards (using approx. gradients)
  - Examples: AlphaGo & modern Atari agents
- Value-based deep RL
  - Basically value iteration, but using a deep neural network (= function approximator) to generalize across many states and actions
  - Approximate optimal state-value function $U(s)$ or state-action value function $Q(s, a)$
- Model-based deep RL
  - If transition model T is not known
  - Approximate T with a deep neural network (learned from data)
  - Plan using this approximate transition model

$\rightarrow$ Use deep neural networks to represent policy / value function / model

Lecture Overview

Deep Learning Focuses on Perception

- Excellent results for perception tasks from raw data
  - Computer vision (from raw pixels)
  - Speech recognition (from raw audio)
  - Text recognition (from raw characters)
  - ...

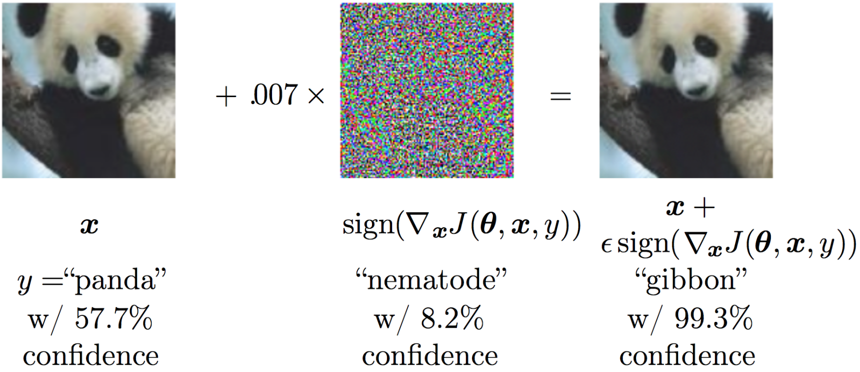
Deep Learning Focuses on Perception

- Excellent results for perception tasks from raw data
  - Computer vision (from raw pixels)
  - Speech recognition (from raw audio)
  - Text recognition (from raw characters)
  - ...
- But all of this is bottom-up
  - No top-down reasoning
  - No logic, planning, etc.
  - Although there are some modern works on memory mechanisms, attention, etc.

Deep Learning Focuses on Perception

- Excellent results for perception tasks from raw data
  - Computer vision (from raw pixels)
  - Speech recognition (from raw audio)
  - Text recognition (from raw characters)
  - ...
- But all of this is bottom-up
  - No top-down reasoning
  - No logic, planning, etc.
  - Although there are some modern works on memory mechanisms, attention, etc.
- Deep networks can be combined with more traditional methods
  - E.g., AlphaGo: combination with Monte Carlo Tree Search (MCTS)
  - Some work on combining logic with deep learning

Adversarial examples: we're very far from human-level performance



- Even for very strong networks we can find adversarial examples
  - By following the gradient of the cost function w.r.t the input

Lecture Overview

Summary: Why is Deep Learning so Popular?

- Excellent empirical results in many domains
  - very scalable to big data
  - but beware: not a silver bullet

Summary: Why is Deep Learning so Popular?

- Excellent empirical results in many domains
  - very scalable to big data
  - but beware: not a silver bullet
- Analogy to the ways humans process information
  - mostly tangential

Summary: Why is Deep Learning so Popular?

- Excellent empirical results in many domains
  - very scalable to big data
  - but beware: not a silver bullet
- Analogy to the ways humans process information
  - mostly tangential
- Allows end-to-end learning
  - no more need for many complicated subsystems
  - e.g., dramatically simplified Google's translation pipeline

Summary: Why is Deep Learning so Popular?

- Excellent empirical results in many domains
  - very scalable to big data
  - but beware: not a silver bullet
- Analogy to the ways humans process information
  - mostly tangential
- Allows end-to-end learning
  - no more need for many complicated subsystems
  - e.g., dramatically simplified Google's translation pipeline
- Very versatile/flexible
  - easy to combine building blocks
  - allows supervised, unsupervised, and reinforcement learning

Lots of Work on Deep Learning in Freiburg

- Computer Vision (Thomas Brox)
  - Images, video
- Robotics (Wolfram Burgard)
  - Navigation, grasping, object recognition
- Neurorobotics (Joschka Boedecker)
  - Robotic control
- Machine Learning (Frank Hutter)
  - Foundations: optimization, neural architecture search, learning to learn
- Neuroscience (Tonio Ball, Michael Tangermann, and others)
  - EEG data and other applications from BrainLinks-BrainTools

→ Details when the individual groups present their research

Summary by learning goals

Having heard this lecture, you can now . . .

- Explain the terms representation learning and deep learning
- Explain why deep learning is so popular
- Describe the main principles behind MLPs
- Discuss some limitations of deep learning
- On a high level, describe
  - Convolutional Neural Networks
  - Recurrent Neural Networks
  - Deep Reinforcement Learning