COGS 181, Fall 2017

Neural Networks and Deep Learning

Zhuowen Tu

Department of Cognitive Science
University of California, San Diego

Syllabus

Syllabus

Lecture Time:

Tuesday and Thursday, 12:30p-1:50pm CSB 001

Lab hours:

Wednesday 4:00p-4:50p CSB 005
Friday 1:00p-1:50p CSB 005

TA:

Zeyu Chen (zec003@eng.ucsd.edu)
Long Jin (longjin@eng.ucsd.edu)

IA:

Adilijiang Ainihaer (aainihae@ucsd.edu)

IA:

Text Books:
This course will be self-contained but below are some useful books that are good to read:

Deep Learning (Ian Goodfellow and Yoshua Bengio and Aaron Courville)
Neural Networks for Machine Learning (Geoffrey Hinton)

R. Duda, P. Hart, D. Stork, "Pattern Classification", second edition, 2000.

Web Resources:

Piazza: **https://piazza.com/configure-classes/fall2017/cogs181**
Ted: ted.ucsd.edu
Podcast: https://podcast.ucsd.edu/

Office Hours:

Zhuowen Tu, Tuesday and Thursday 2:00-3:00pm, CSB 132

Course Description:

Recent developments in deep neural networks approaches have led to a significant boost to state-of-the-art learning systems in a wide range of domains including machine learning, robotics, perception, computer vision, artificial intelligence, speech recognition, neural computation, medical imaging, bio-informatics, computational linguistics, and social data analysis. This course will cover the basics about neural networks, as well as recent developments in deep learning including deep belief nets, convolutional neural networks, recurrent neural networks, long-short term memory, and reinforcement learning. We will study details of the deep learning architectures with a focus on learning end-to-end models for these tasks, particularly image classification. Students will learn hands-on skills to do cutting-edge research by implementing, training and debugging neural networks.

Prerequisites:

Mathematics 20F (Linear Algebra) or Mathematics 31AH (Honors Linear Algebra), and Mathematics 180A (Introduction to Probability) or ECE 109 (Engineering Probability & Statistics), and COGS 109 (Modeling and Data Analysis) or COGS 108 or CSE 11 (Introduction to Computer Science & Object-Oriented Programming: Java), or consent of instructor.

Grading policy:
Assignments: 40% (Late policy: 5% reduction for the first day past due and every 10% reduction afterwards).
Midterms: 40%
Final project: 20% (Late policy: 5% reduction for the first day past due and every 10% reduction afterwards.).
Bonus points: 3% (classroom participation and final project)

Policy on Integrity of Scholarship

TA:

Zeyu Chen

Long Jin

IA:

Adilijiang Ainihaer

Grading Policy

Assignments: 40%

Midterm exams: 40%

Final project: 20%

Bonus points: 3% (classroom participation + final project)

Course Webpage

https://**sites.google.com/site/ucsdcogs181fall2017/**

Piazza

https://piazza.com/ucsd/fall2017/cogs181/

Ted: ted.ucsd.edu

Class schedule

Calendar and Class Notes

Date and Lecture #	Topic	Readings
Week 1, 09/28/17 (Thursday)	Course overview and introduction Concepts about learning and neural networks	Probability theory (by Matthew Shum) Introduction to probability by C.M. Grinstead and J.L. Snell Linear algebra review A few useful things to know about machine learning(Pedro Domingos) Learning Python Why do we need machine learning? What are neural networks? (G. Hinton)
Week 2 10/03/17 (Tuesday)	Data Classification Discriminative vs Generative learning	Deep Learning: Linear Algebra Deep Learning: Machine Learning Basics
Week 2 10/05/17 (Thursday)	Decision boundaries Introduction to neurons and neural networks	Deep Learning: Numeric Optimization
Week 3 10/10/17 (Tuesday)	Error metrics and optimization Least square estimation Convexity	Precision and recall Receive operating characteristic Least square estimation Convex function Matrix calculus
Week 3 10/12/17 (Thursday)	Gradient descent and logistic regression Perceptron: training and testing	Gradient descent Logistic regression Perceptron
Week 4 10/17/17 (Tuesday)	Hopfiled neural networks	Hopfield Neural Network J. J. Hopfield and D. W. Tank, "Neural" computation of decisions in optimization problems, Biological Cybernetics 1985.
Week 4 10/19/17 (Thursday)	Midterm 1 Hopfield neural networks	
Week 5 10/24/17 (Tuesday)	Feedforward networks	
Week 5 10/26/17 (Thursday)	Feedforward networks	Feedforwad neural networks Backpropagation
Week 6 10/31/17 (Tuesday)	Convolutional neural networks	Convolution
Week 6 11/02/17 (Thursday)	Convolutional neural networks: learning strategies, applications	Circuit illustration of backpropagation
Week 7 11/07/17 (Tuesday)	Convolutional neural networks	
Week 7 11/09/17 (Thursday)	Modern CNN architectures	AlexNet GoogLeNet VGG ResNet
Week 8 11/14/17 (Tuesday)	Deep belief nets	Boltzman machine Restricted Boltzman manchine
Week 8 11/16/17 (Thursday)	Midterm 2 Auto-encoder	
Week 9 11/21/17 (Tuesday)	Language modeling and Word2Vec Recurrent neural networks	Word2Vec
Week 9 11/23/17 (Thursday)	No class	Recurrent neural network
Week 10 12/05/17 (Tuesday)	Long-short memory	Long short term memory Gated recurrent unit
Week 10 12/07/17 (Thursday)	Deep reinforcement learning	

Background materials to learn and prepare

Linear algebra review:

http://cs229.stanford.edu/section/cs229-linalg.pdf

Probability theory

http://people.hss.caltech.edu/~mshum/stats/lect1.pdf

Python/IPython

http://ipython.org/

http://cs231n.github.io/python-numpy-tutorial/

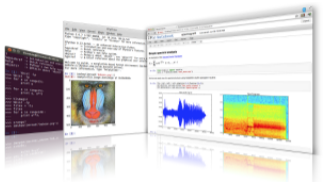
IPython Notebook

IP[y]: IPython
Interactive Computing

Install · Documentation · Project · Jupyter · News · Cite · Donate · Books

IPython provides a rich architecture for interactive computing with:

- A powerful interactive shell.
- A kernel for Jupyter.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into your own projects.
- Easy to use, high performance tools for parallel computing.



To get started with IPython in the Jupyter Notebook, see our official example collection. Our notebook gallery is an excellent way to see the many things you can do with IPython while learning about a variety of topics, from basic programming to advanced statistics or quantum mechanics.

To learn more about IPython, you can download our talks and presentations, or read our extensive documentation. IPython is open source (BSD license), and is used by a range of other projects; add your project to that list if it uses IPython as a library, and please don't forget to cite the project.

IPython supports Python 2.7 and 3.3 or newer. Our older 1.x series supports Python 2.6 and 3.2.

Jupyter and the future of IPython

IPython is a growing project, with increasingly language-agnostic components. IPython 3.x was the last monolithic release of IPython, containing the notebook server, qtconsole, etc. As of IPython 4.0, the language-agnostic parts of the project: the notebook format, message protocol, qtconsole, notebook web application, etc. have moved to new projects under the name Jupyter. IPython itself is focused on interactive Python, part of which is providing a Python kernel for Jupyter.

Announcements

- **IPython 6.1**, and **IPython 5.4**: These two releases were made on May 31st 2017.
- **IPython 6.0**: This release, the first to require Python 3, integrates the Jedi library for completion. See the release notes for more information about what's new.
- **JupyterCon 2017**: The first Jupyter Community Conference will take place in New York City on August 23-25 2017, along with a satellite training program on August 22-23. The Project Jupyter team has partnered with O'Reilly Media for this event; for more details, including submitting a talk, see the JupyterCon website.
- **IPython 5.0**: The release of IPython 5.0 brings a major revision of the terminal interface, including syntax highlighting as you type and better multiline editing, thanks to the `prompt_toolkit` library. See the release notes for more about the new features.
- **Book**: Cyrille Rossant has published the second edition of the IPython minibook: Learning IPython for Interactive Computing and Data Visualization, for which Damian Avila was a technical reviewer. We thank Packt Publishing for donating a portion of the proceeds from this book to support IPython's development.
- **O'Reilly Book**: Mining the Social Web is an open source data science project and book that features nearly 130 examples with IPython Notebook and a Vagrant-powered virtual machine environment. You can preview all of the example notebooks from its GitHub repository on IPython's Notebook Viewer here.

More news...

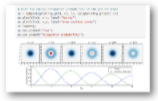
Google Custom Se Search

JUPYTERCON

jupytercon

NOTEBOOK VIEWER

Share your notebooks



COMMUNITY

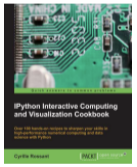
Stack Overflow
Mailing list
File a bug
Reddit

Tweet

FOR DEVELOPERS

Mailing list
Development Chat Room
Development information
Travis CI
Wiki

BOOK



Support IPython

Donate Now

Find out more...

IP[y]: IPython
Interactive Computing

Install · Documentation · Project · Jupyter · News · Cite · Donate · Books

Installing IPython

There are multiple ways of installing IPython. This page contains simplified installation instructions that should work for most users. Our official documentation contains more detailed instructions for manual installation targeted at advanced users and developers.

If you are looking for installation documentation for the notebook and/or qtconsole, those are now part of Jupyter.

I already have Python

If you already have Python installed and are familiar with installing packages, you can get IPython with **pip**:

```
pip install ipython
```

I am getting started with Python

For new users who want to install a full Python environment for scientific computing and data science, we suggest installing the Anaconda or Canopy Python distributions, which provide Python, IPython and all of its dependences as well as a complete set of open source packages for scientific computing and data science.

1. Download and install Continuum's Anaconda or the free edition of Enthought's Canopy.
2. Update IPython to the current version using the Terminal:

Anaconda:

```
conda update conda
conda update ipython
```

Enthought Canopy:

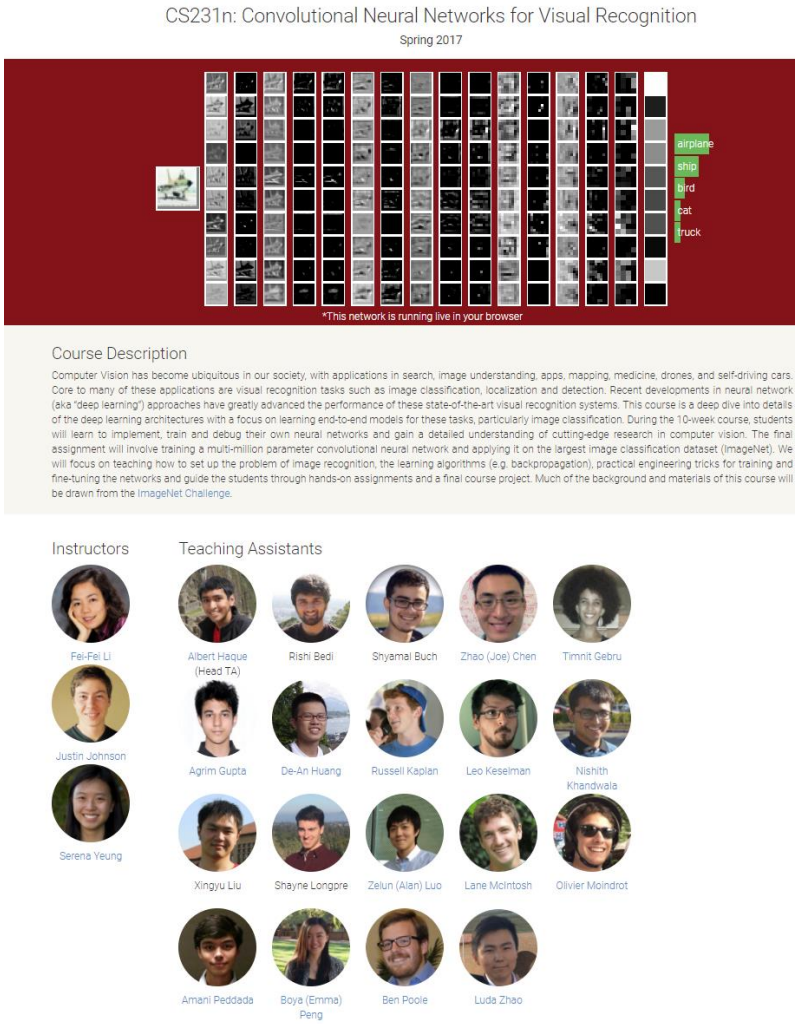
```
enpkg ipython
```

Downloads

You can manually download IPython from GitHub or PyPI. To install one of these versions, unpack it and run the following from the top-level source directory using the Terminal:

```
pip install .
```

Stanford CS231n: http://cs231n.stanford.edu/



Lecture videos: https://www.youtube.com/playlist?list=PL3FW7Lu3i5JvHM8ljYj-zLfQRF3EO8sYv

IPython tutorial to quickly get started

CS231n Convolutional Neural Networks for Visual Recognition

Python Numpy Tutorial

This tutorial was contributed by Justin Johnson.

We will use the Python programming language for all assignments in this course. Python is a great general-purpose programming language on its own, but with the help of a few popular libraries (numpy, scipy, matplotlib) it becomes a powerful environment for scientific computing.

We expect that many of you will have some experience with Python and numpy; for the rest of you, this section will serve as a quick crash course both on the Python programming language and on the use of Python for scientific computing.

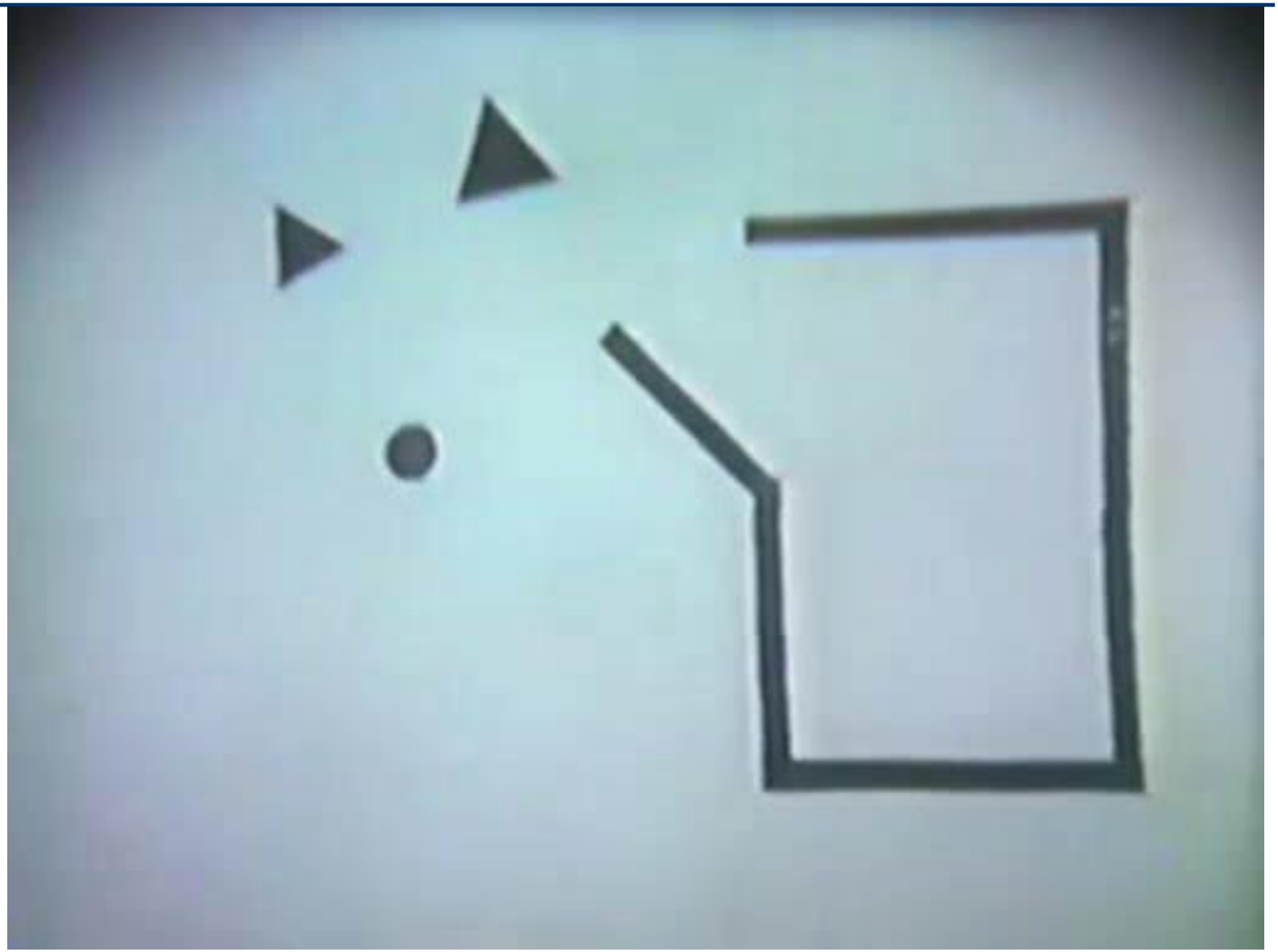
Some of you may have previous knowledge in Matlab, in which case we also recommend the numpy for Matlab users page.

You can also find an IPython notebook version of this tutorial here created by Volodymyr Kuleshov and Isaac Caswell for CS 228.

Table of contents:

Python

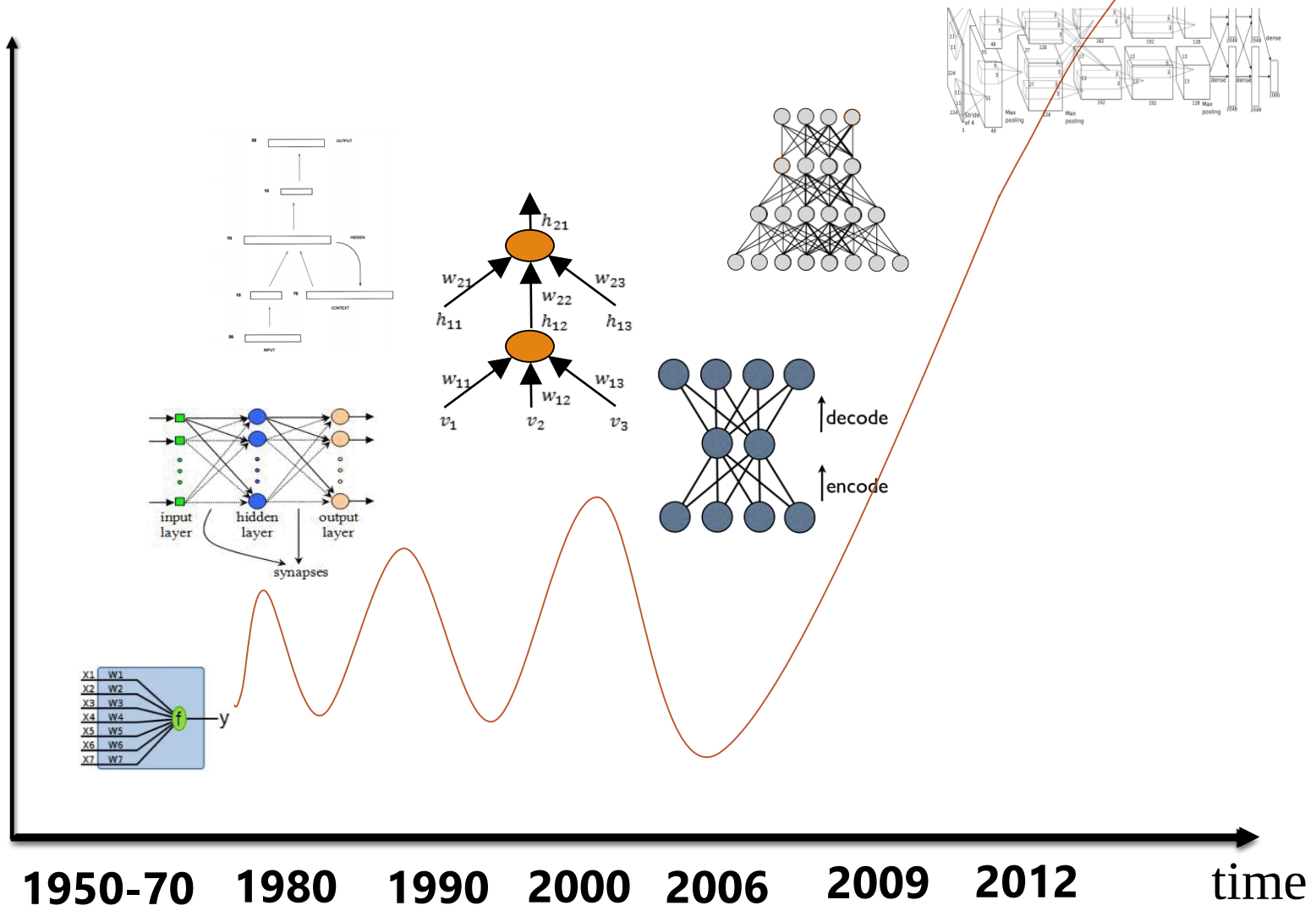
Why is representation so important?



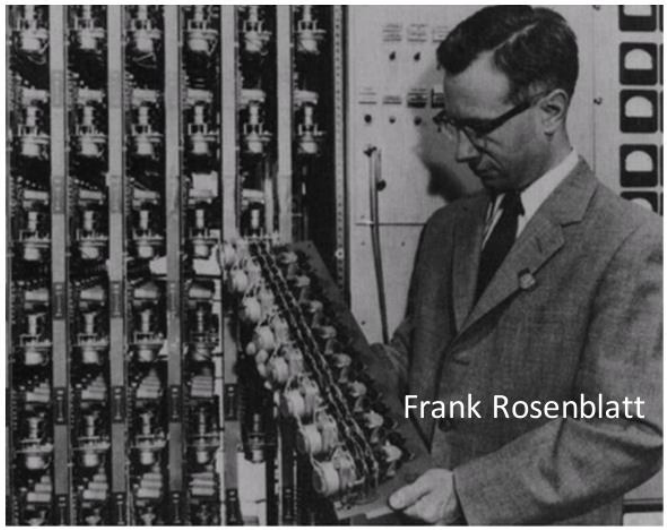
Video： Apparent Behavior, Heider and Simmel, 1944

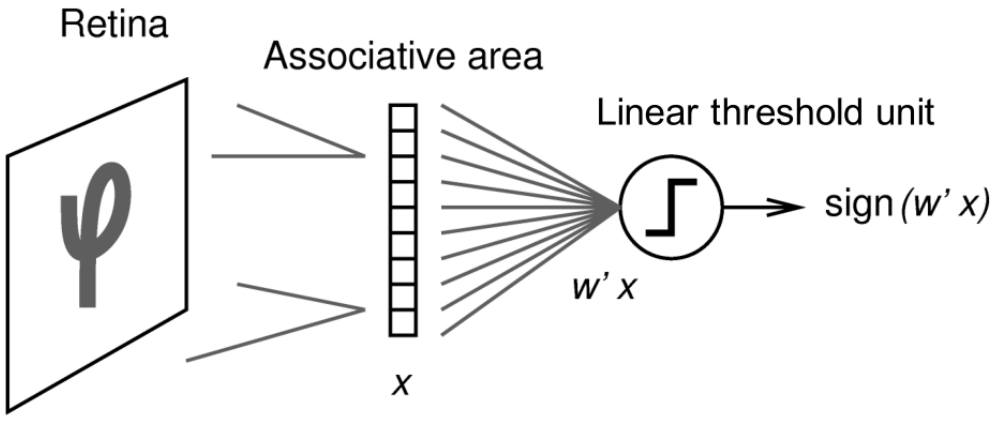
Artificial neural networks: a brief history

Rosenblatt, F. (1958). "The Perceptron: A Probalistic Model For Information Storage And Organization In The Brain".
Hopfield J. (1982), "Neural networks and physical systems with emergent collective computational abilities", PNAS.
Rumelhart D., Hinton G. E., Williams R. J. (1986), "Learning internal representations by error-propagation".
Elman, J.L. (1990). "Finding Structure in Time".
Hinton, G. E.; Osindero, S.; Teh, Y. (2006). "A fast learning algorithm for deep belief nets".
Krizhevsky, A.; Sutskever, I.; Hinton, G. (2012) “ImageNet Classification with Deep Convolutional Neural Networks”



Perceptron





Supervised learning of the weights w using the Perceptron algorithm.

The XOR problem that kills the Perceptron algorithm

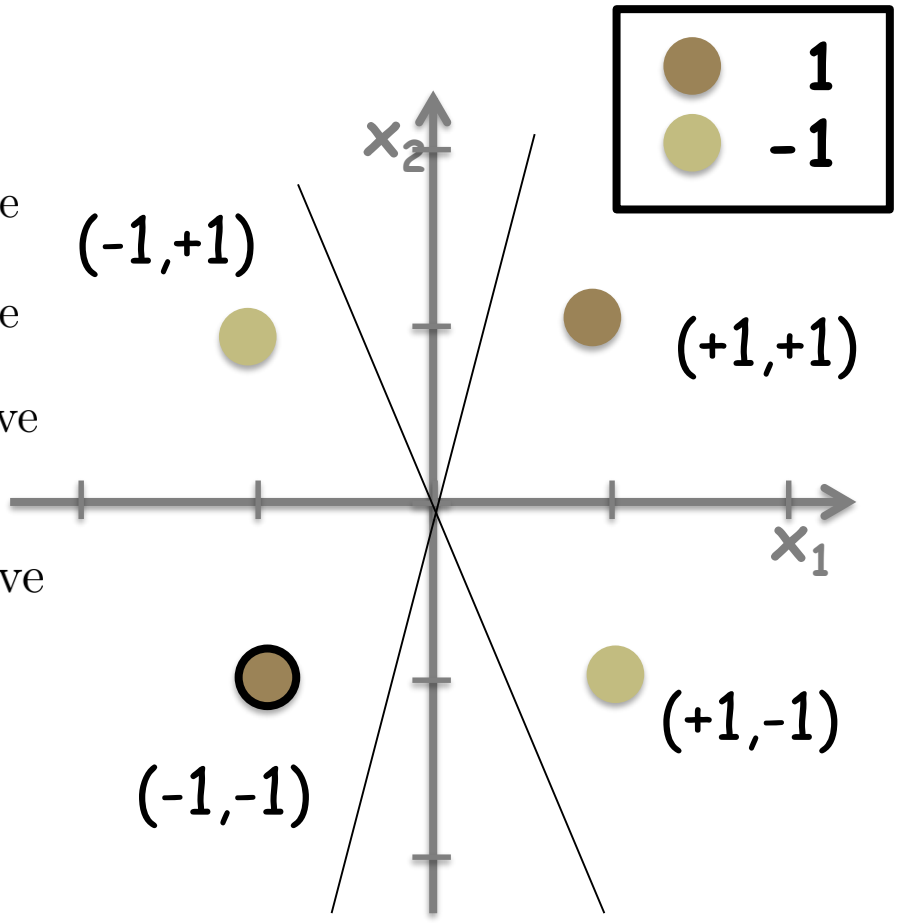
The XOR operator: $\bigoplus$

Positive (+1) $\bigoplus$ Negative (-1) = Negative

Positive (+1) $\bigoplus$ Positive (+1) = Positive

Negative (-1) $\bigoplus$ Positive (+1) = Negative

Neative (-1) $\bigoplus$ Negative (-1) = Positive



AI after Minsky's book *Perceptron*

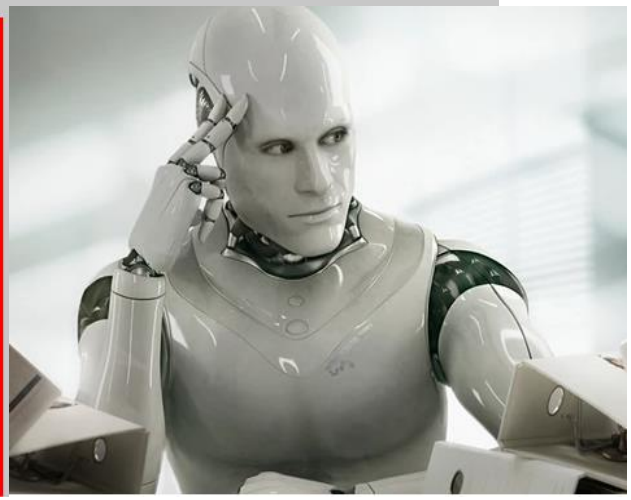
Applications of deep learning technologies

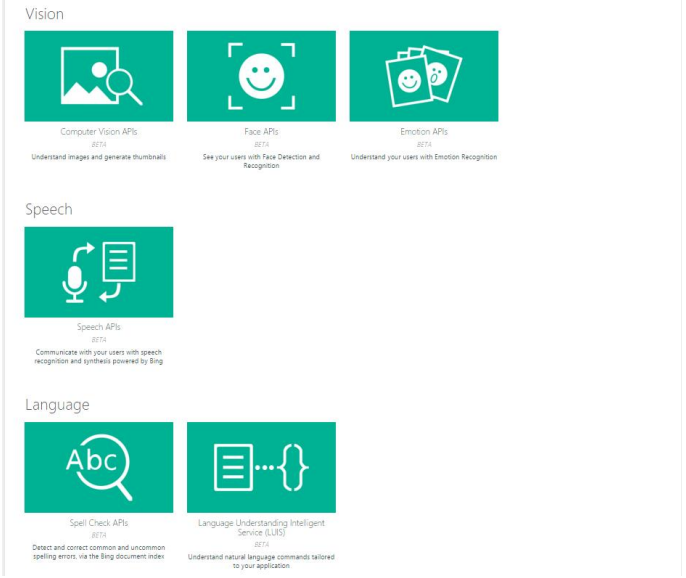
MIT Technology Review

10 BREAKTHROUGH TECHNOLOGIES 2013

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.

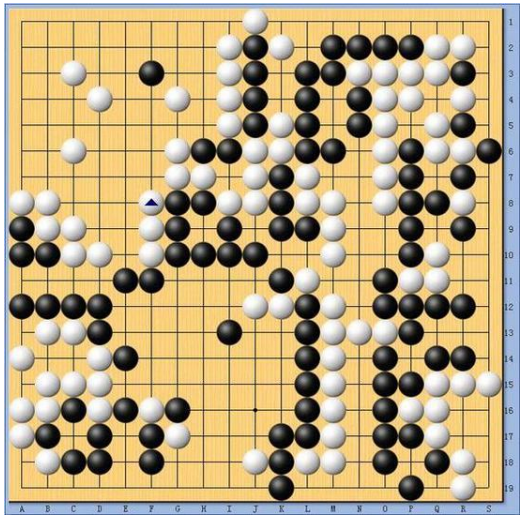




AlphaGo vs. Lee Sedol

AlphaGo

Lee Sedol



2016: match 1
AlphaGo: white; Lee Sedol: black

2017: AlphaGo vs. Jie Ke

Comparison to the 2016 version:

1. **Trained (reinforcement learning) from scratch without human knowledge.**
2. 10 times stronger than the 2016 version (can give up three moves).
3. Run on a single machine (10 TPUs).
4. Consumes 1/10 of the power.

Deep Learning

Geoffrey Hinton started this wave of development of deep learning in 2006.

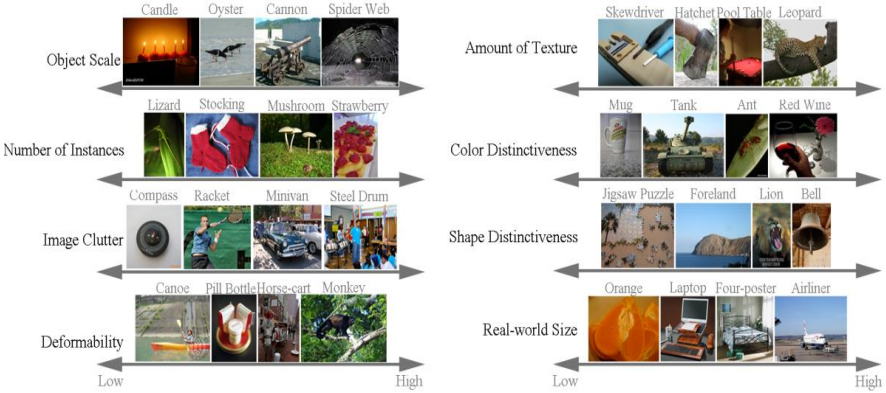
A large number of applications are mainly based on convolutional neural networks based on the models proposed by Yann LeCun.

Yoshua Bengio also made a big contribution in deep learning by proposing many fundamental algorithms.

Factors in the success of deep learning methods

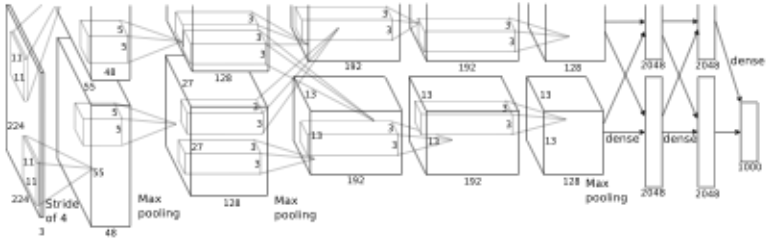
Availability of large amount of training data (big data), e.g. ImageNet.

1.2 million training images, 1,000 classes, 150k test images



Access of modern computing infrastructures, e.g. Nvidia GPUs

New development in neural networks with deep structures, e.g. AlexNet.



"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

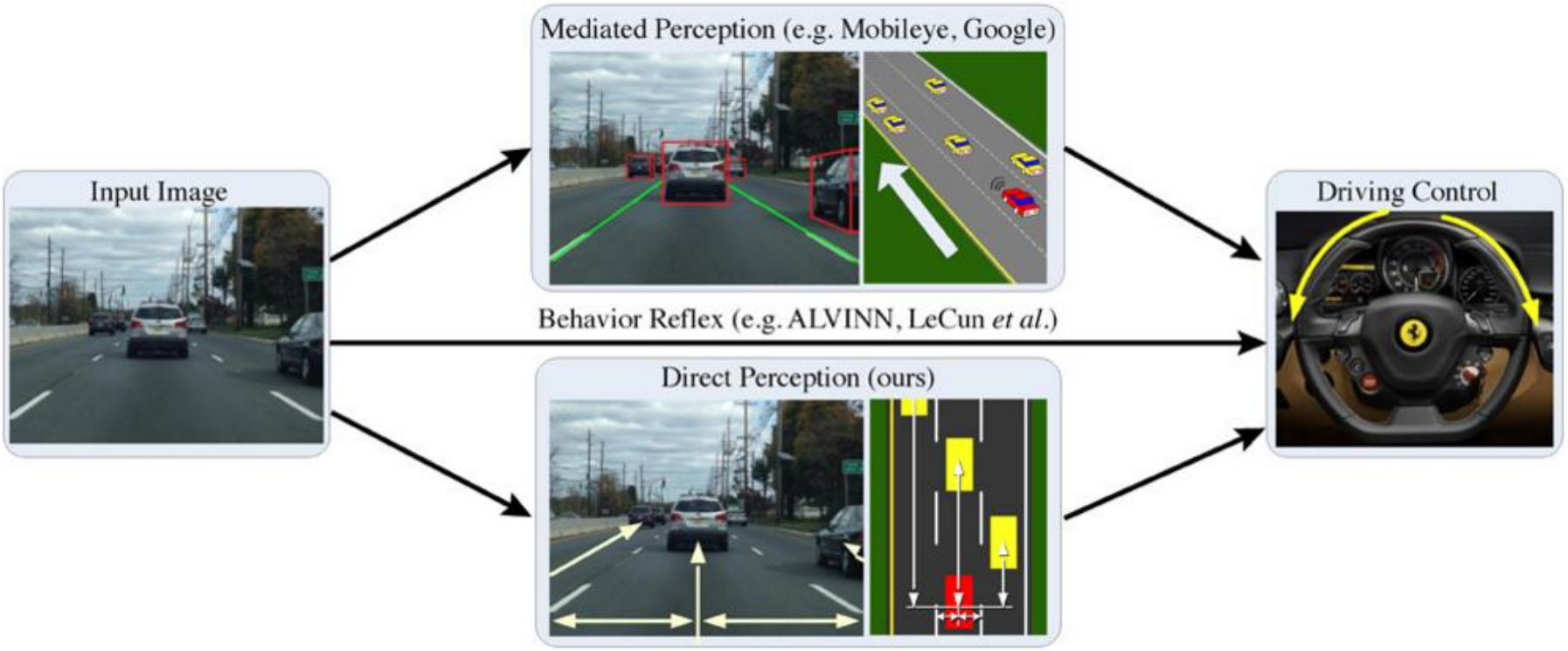
"two young girls are playing with lego toy."

"a young boy is holding a baseball bat."

"a cat is sitting on a couch with a remote control"

"a horse is standing in the middle of a road."

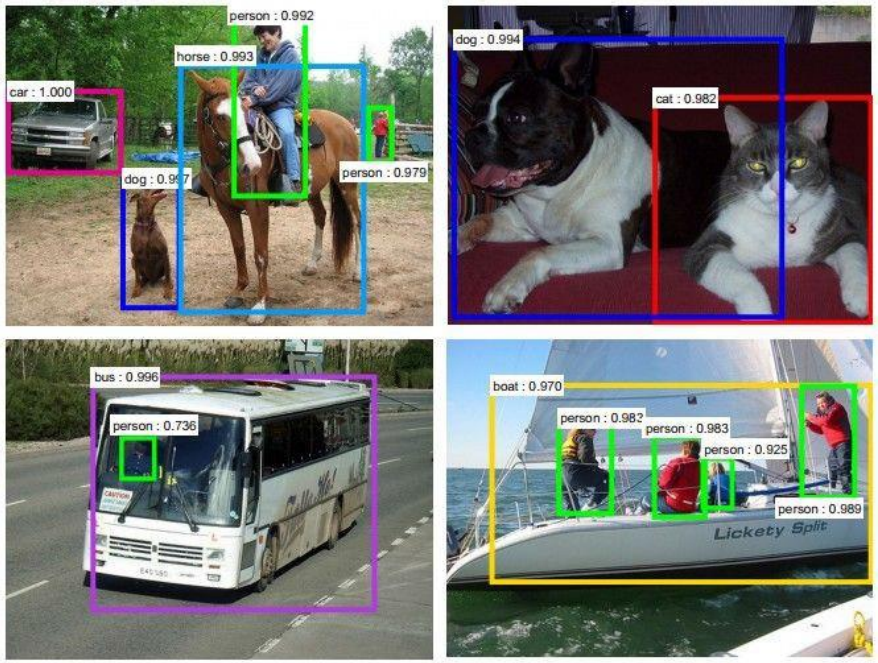
DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving



C. Chen, A. Seff, A. Kornhauser and J. Xiao, ICCV15

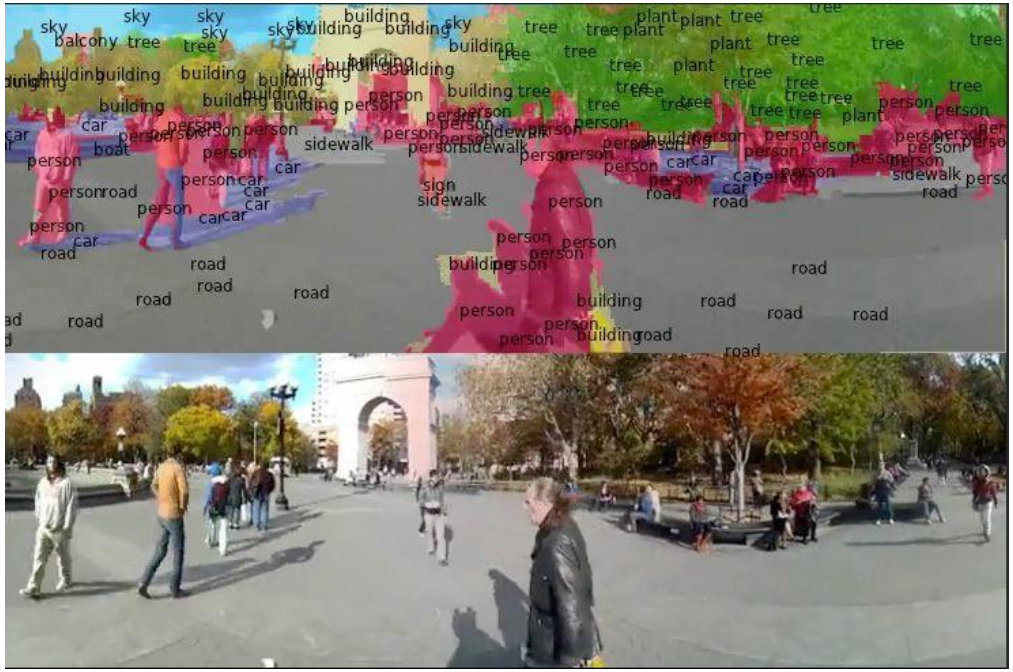
Fast-forward to today: ConvNets are everywhere

Detection



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

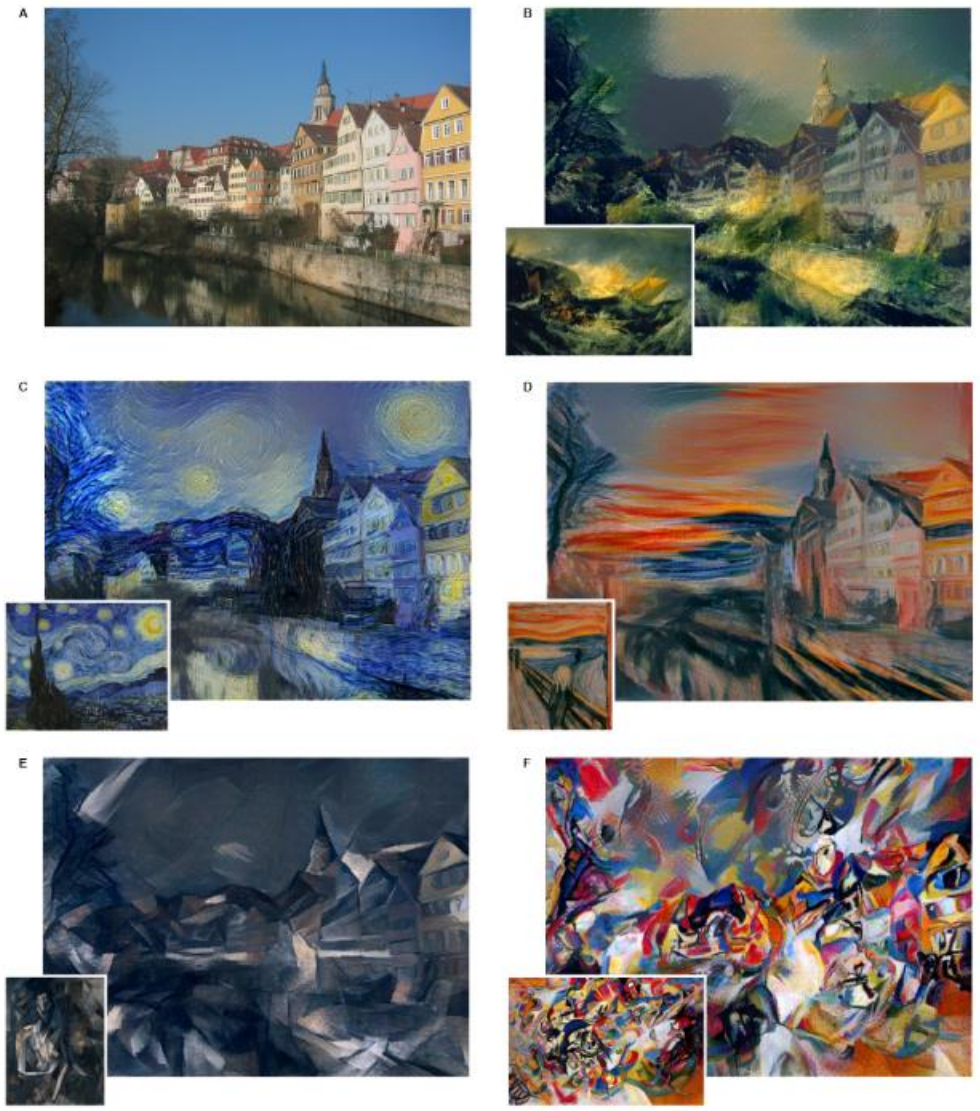
Segmentation



[Farabet et al., 2012]

Neural Artistic Style Learning

Leon A. Gatys, Alexander S. Ecker, Matthias Bethge



Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders, Engel et al. 2017

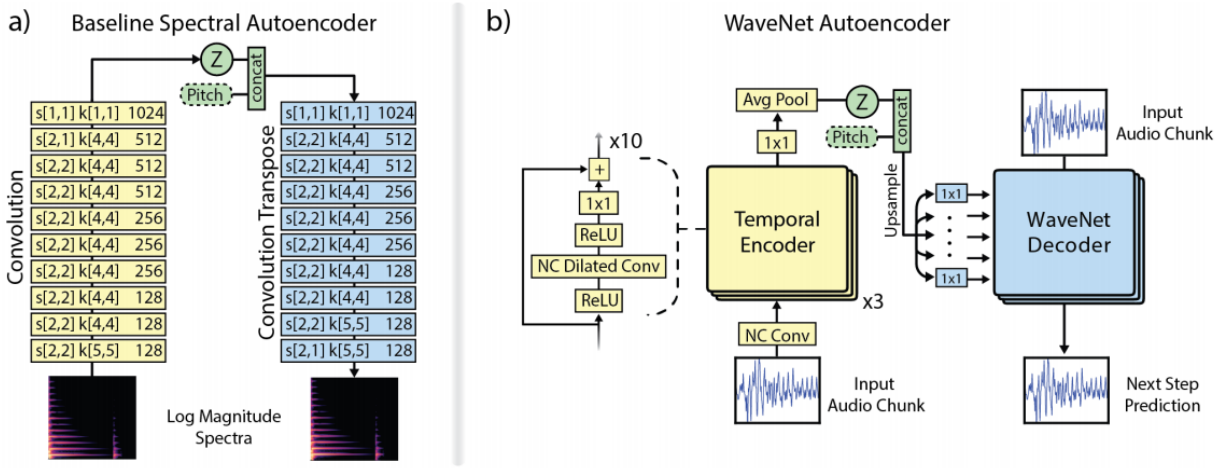
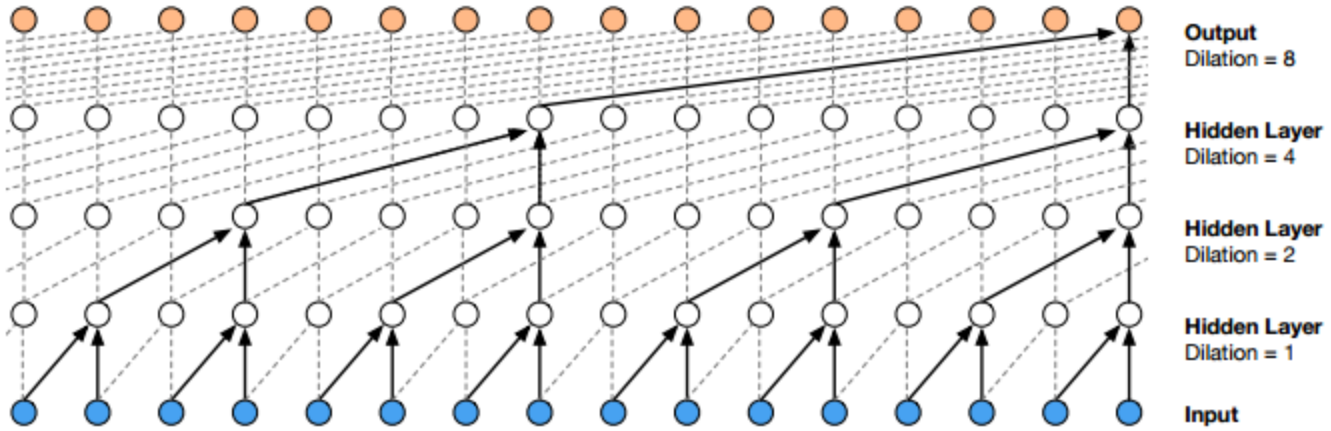


Figure 1. Models considered in this paper. For both models, we optionally condition on pitch by concatenating the hidden embedding with a one-hot pitch representation. 1*a*. Baseline spectral autoencoder: Each block represents a nonlinear 2-D convolution with stride (s), kernel size (k), and channels (#). 1*b*. The WaveNet autoencoder: Downsampling in the encoder occurs only in the average pooling layer. The embeddings are distributed in time and upsampled with nearest neighbor interpolation to the original resolution before biasing each layer of the decoder. 'NC' indicates non-causal convolution. '1x1' indicates a 1-D convolution with kernel size 1. See Section 2.1 for further details.

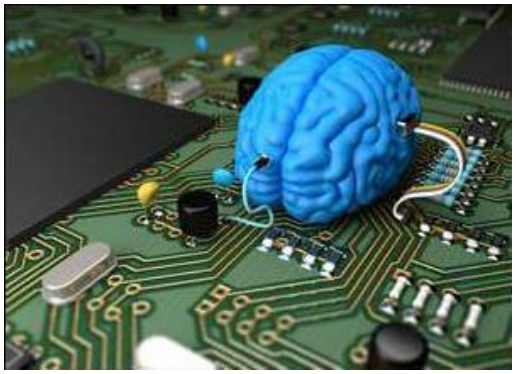
WAVENET: A GENERATIVE MODEL FOR RAW AUDIO, van den Oord et al. 2016



Large-scale Video Classification with Convolutional Neural Networks, Karpathy et al. 2014

From Yann LeCun

What society thinks I do

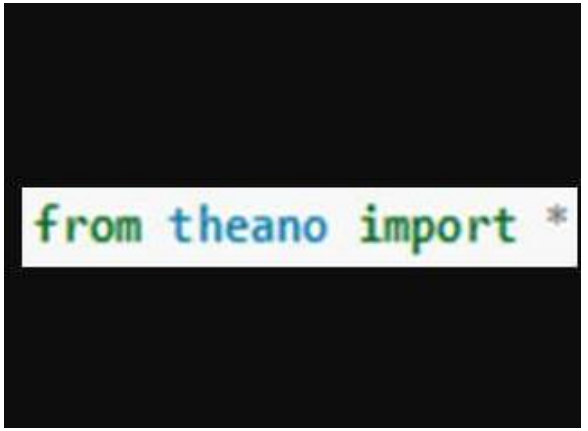


What my friends think I do

What other computer scientists think I do

What mathematicians think I do

What I think I do



What I actually do

A bit of history:

Hubel & Wiesel,

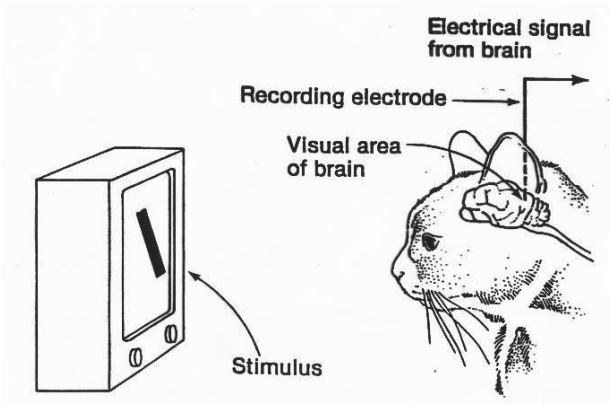
1959

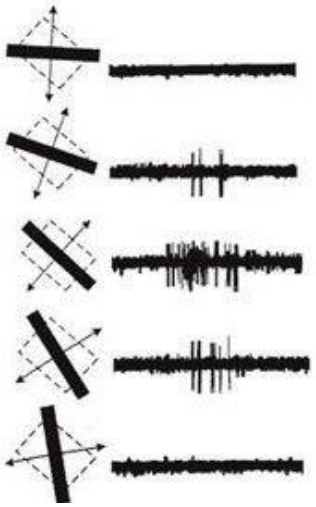
RECEPTIVE FIELDS OF SINGLE NEURONES IN
THE CAT'S STRIATE CORTEX

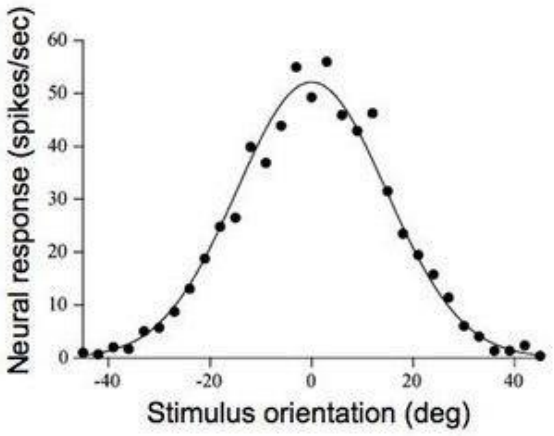
1962

RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX

1968...



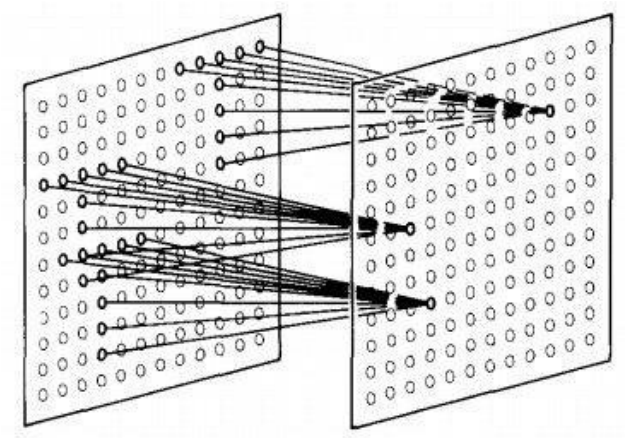


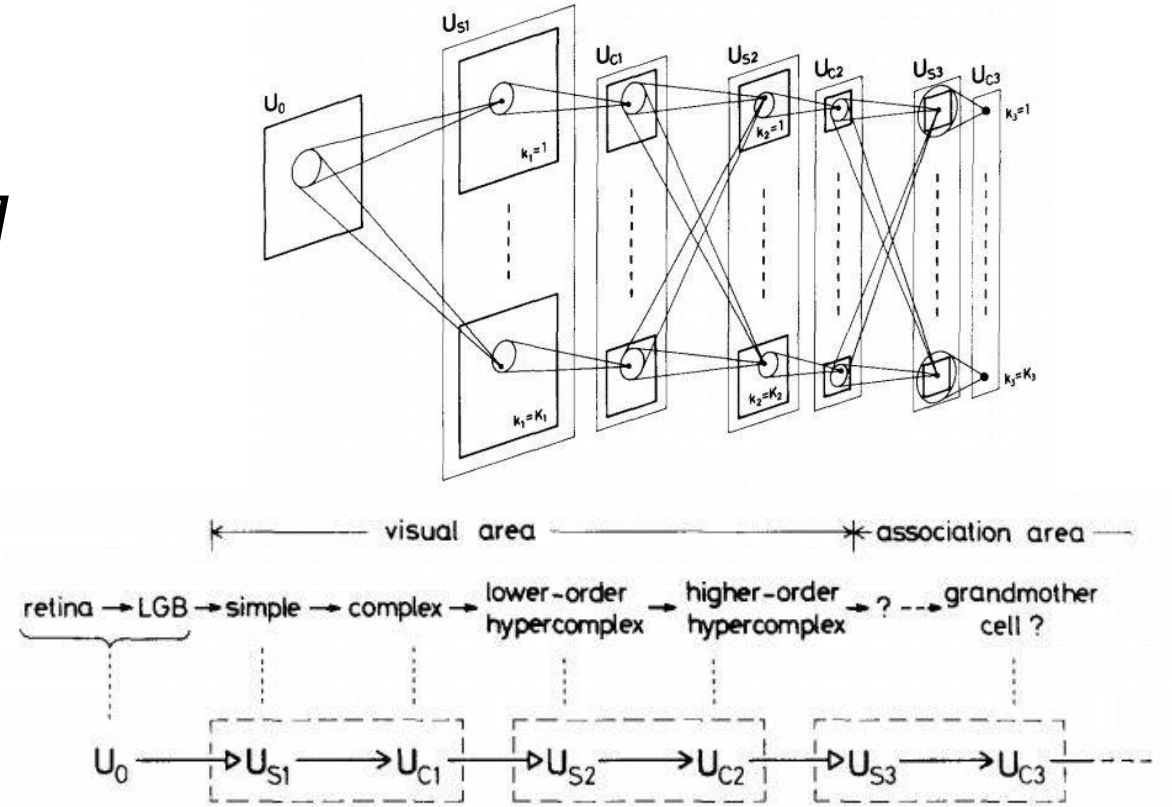


A bit of history:

Neurocognitron
[Fukushima 1980]

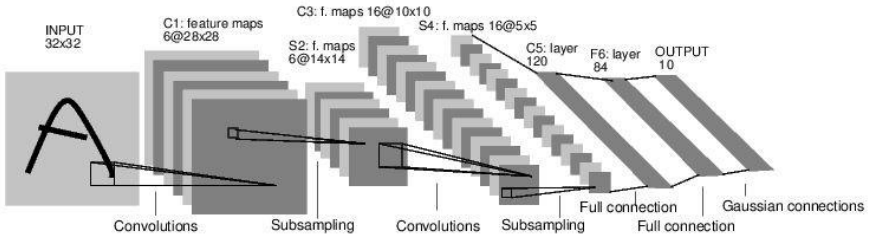
"sandwich" architecture (SCSCSC…)
simple cells: modifiable parameters
complex cells: perform pooling



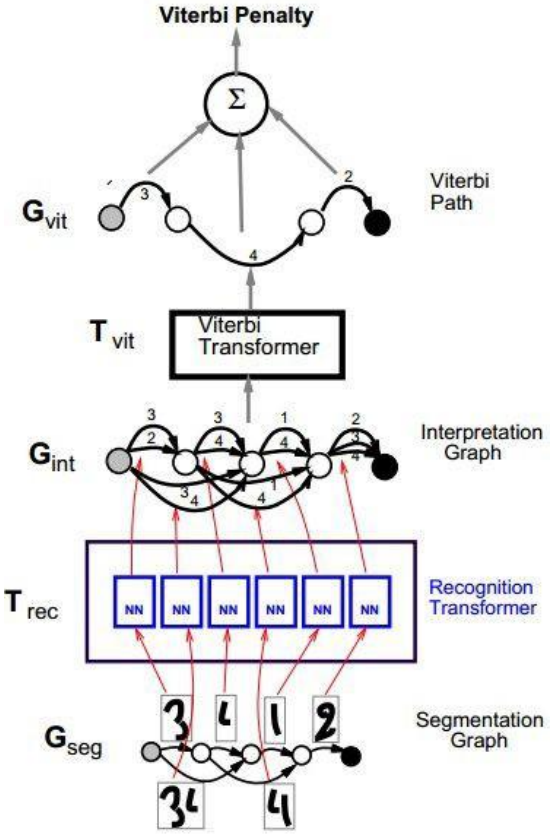


A bit of history:

Gradient-based learning applied to document recognition
[LeCun, Bottou, Bengio, Haffner 1998]

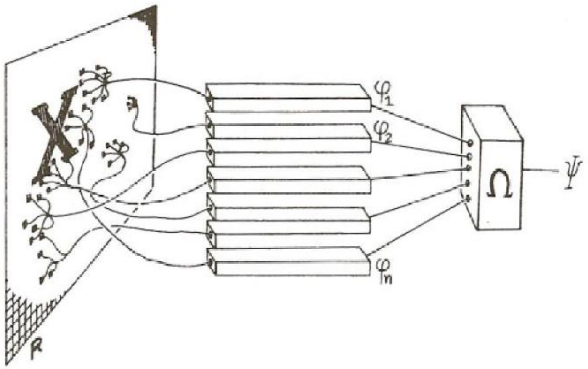


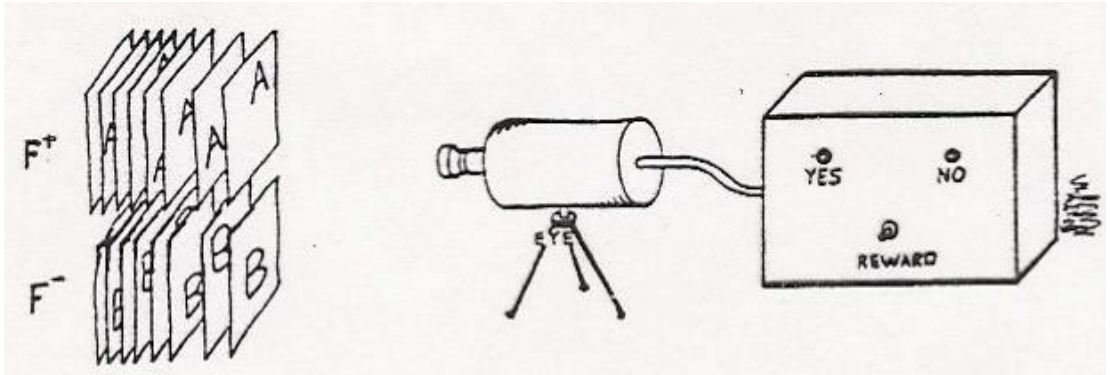
LeNet-5



Artificial neural networks: a brief history

- 1962
  - Frank Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*
  - Perceptron can learn anything you can program it to do.



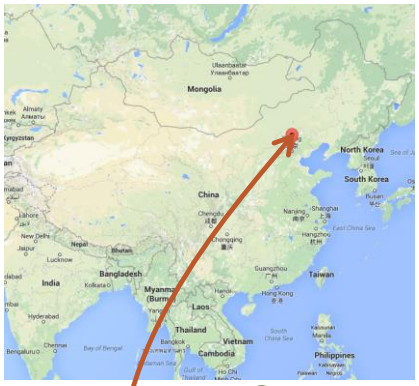


1969

Minsky & Papert, Perceptrons: An introduction to computational geometry

There are many things a perceptron can't in principle learn to do

An illustration

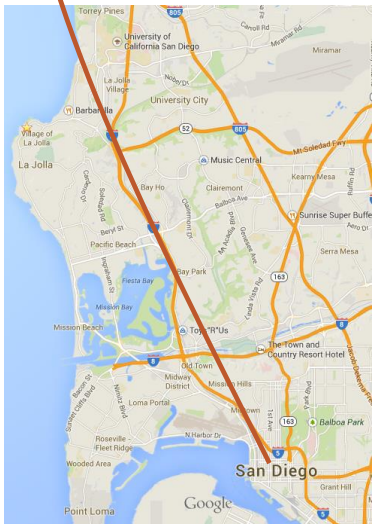


Beijing, China

Our task: to learn from a data (but not always)

representation: vehicles (invention) +

function: itinerary (composition)



San Diego, USA

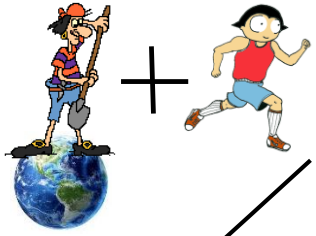
An illustration

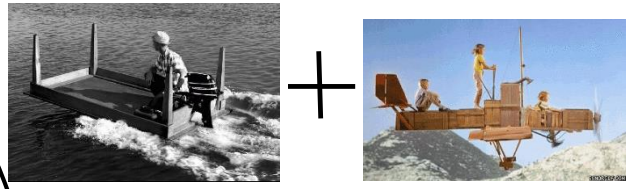
(images obtained by searching Google and Bing)

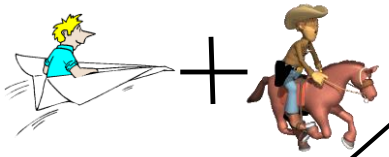
Purpose of a data-driven learning approach:

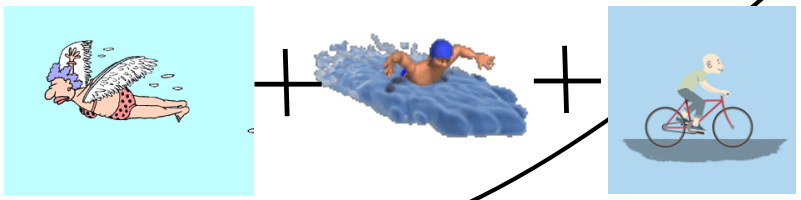
(1) Inventing--features

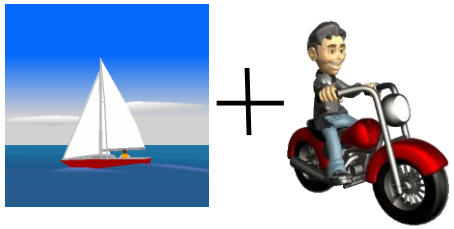
(2) Composing--features

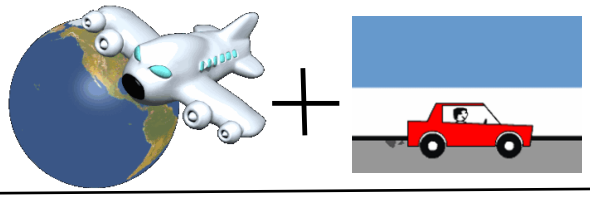








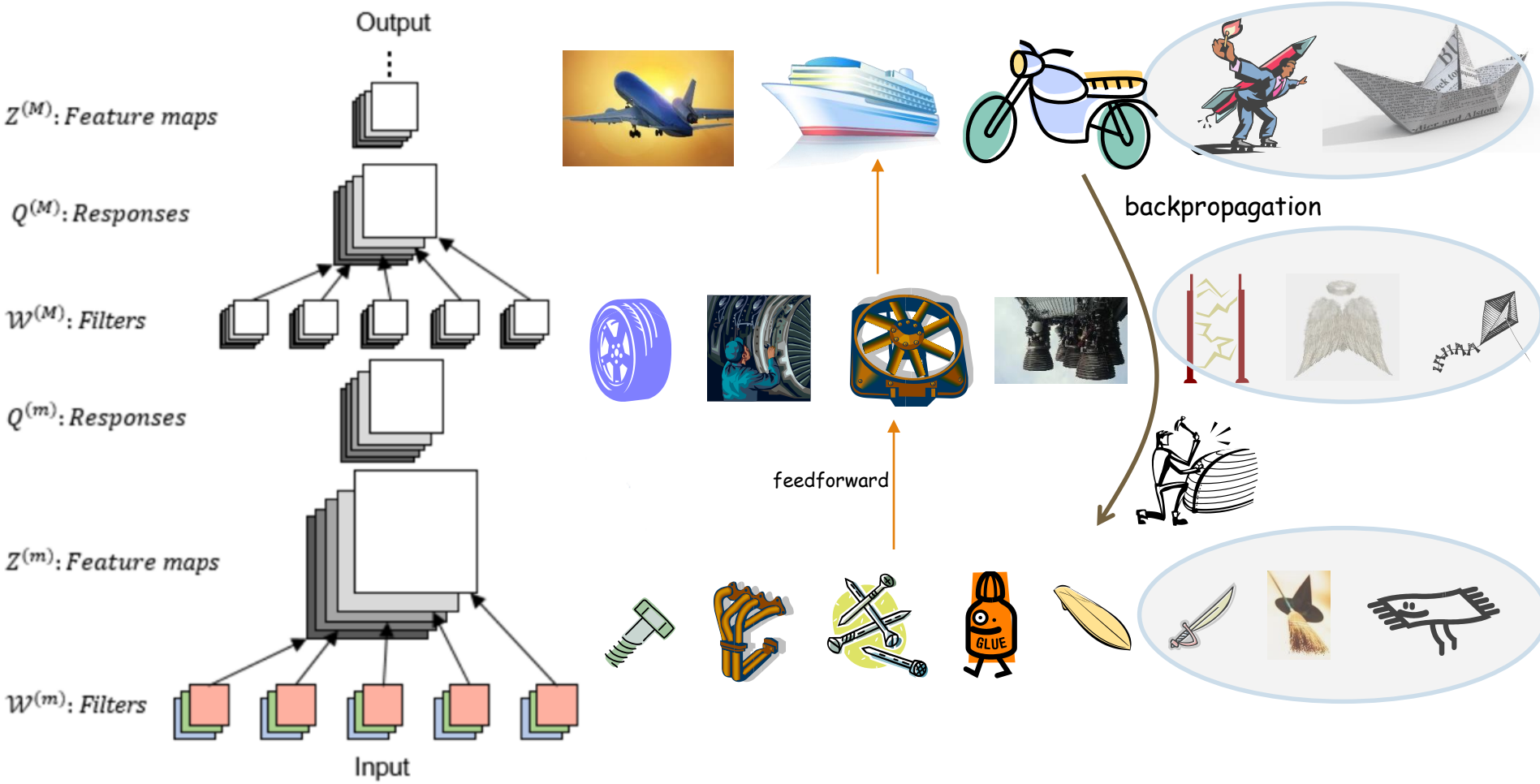




Time

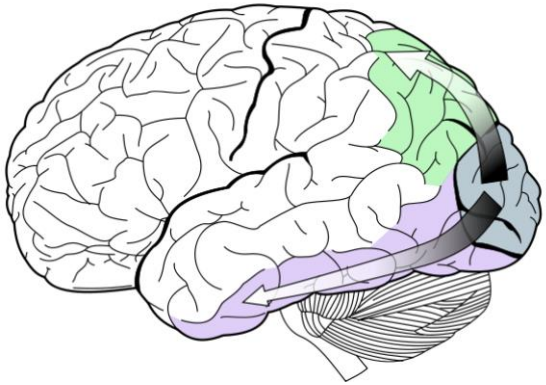
Itinerary

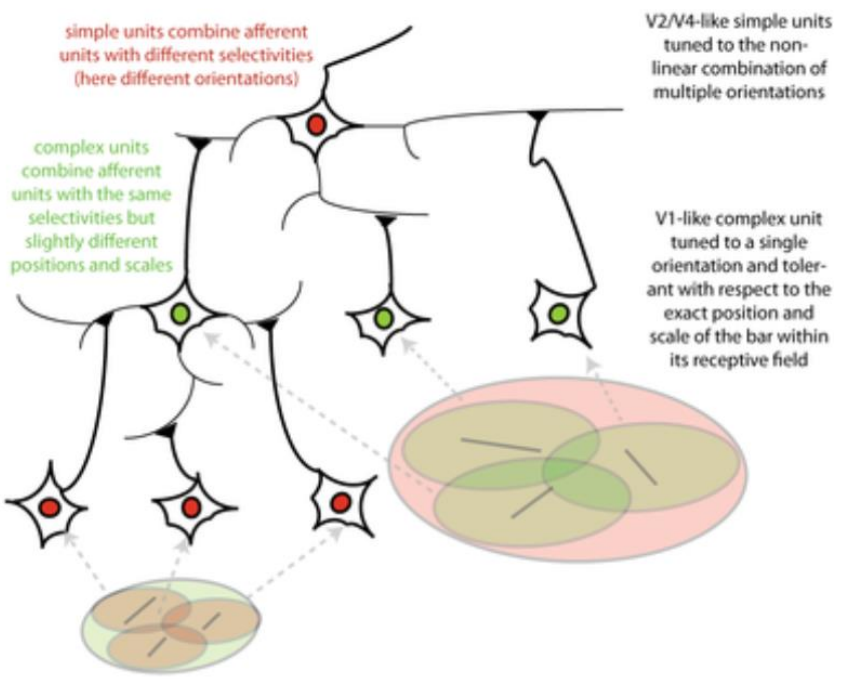
An illustration



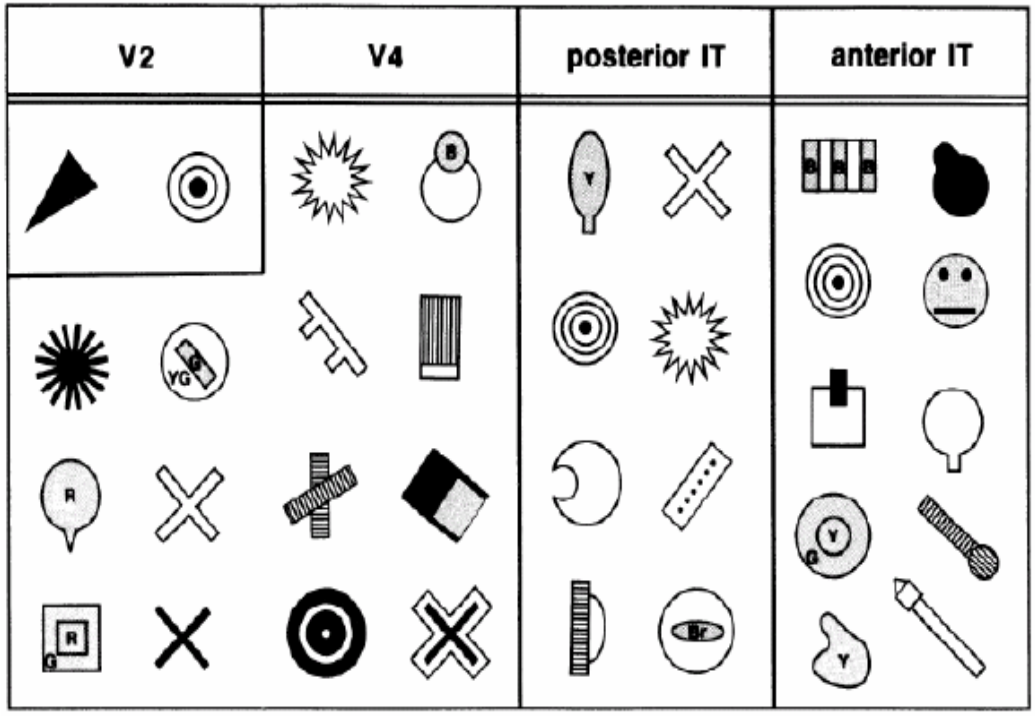
Convolutional neural networks (Y. LeCun)

Visual Representation



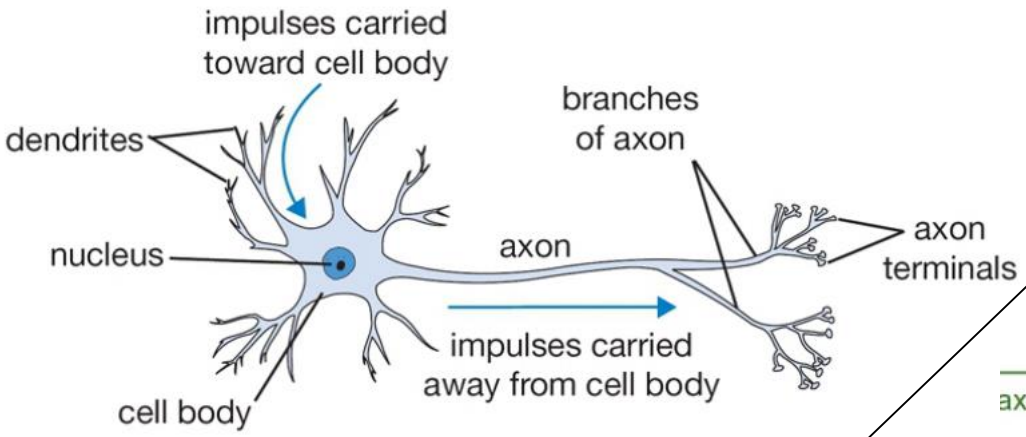


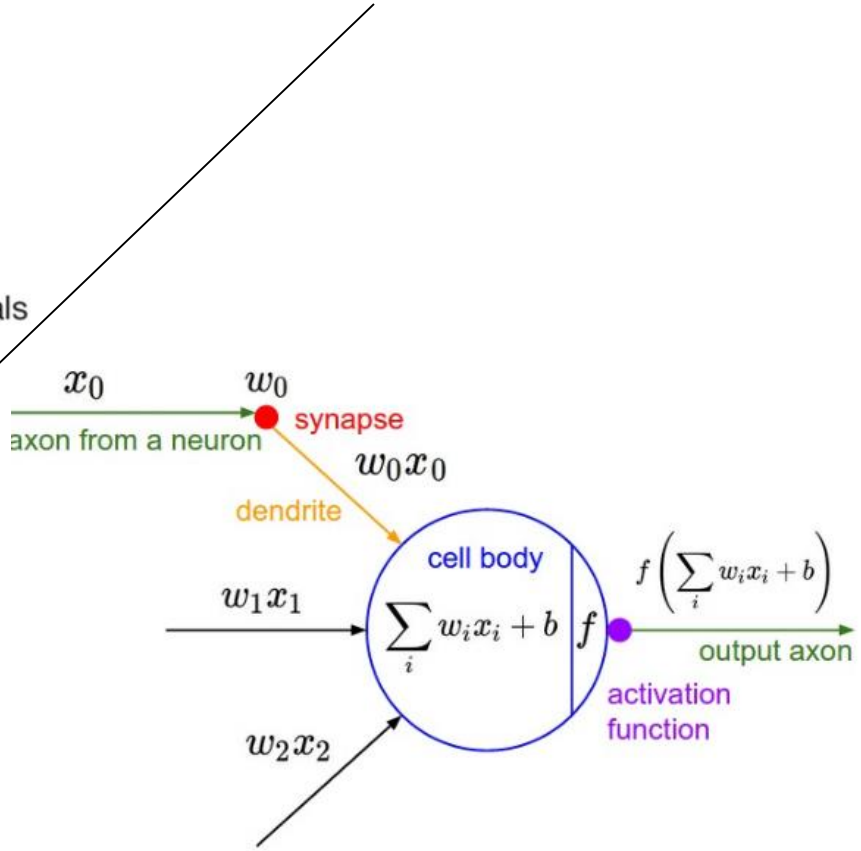
Hubel and Wiesel Model

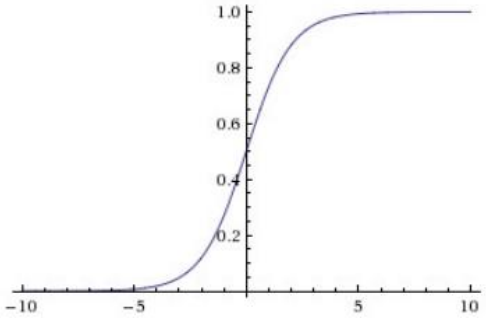


Kobatake and Tanaka, 1994

Perceptron





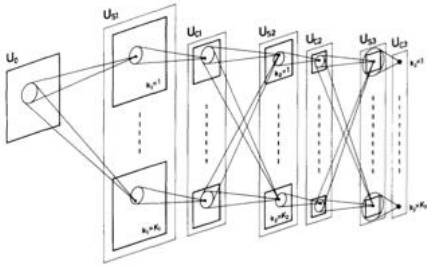


Sigmoid function

$$f(x) = \frac{1}{1+e^{-x}}$$

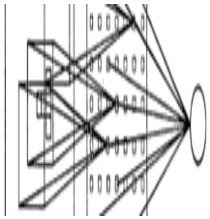
Fei-fei and Karpathy

History of ConvNets



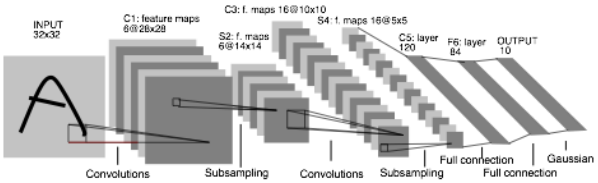
Fukushima 1980

Neocognitron



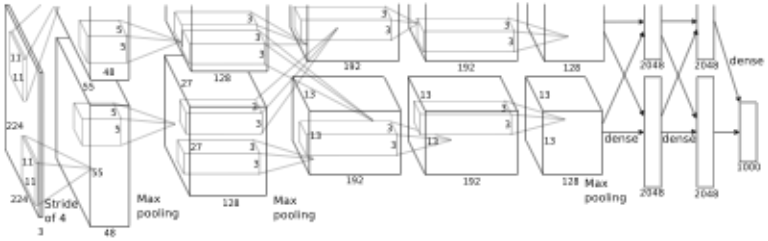
Rumelhart, Hinton, Williams 1986

“T” versus “C” problem



LeCun et al. 1989-1998

Hand-written digit reading



Krizhevksy, Sutskever, Hinton 2012

ImageNet classification

ImageNet experiments