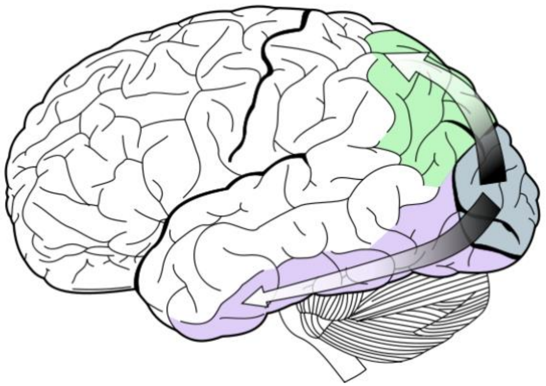
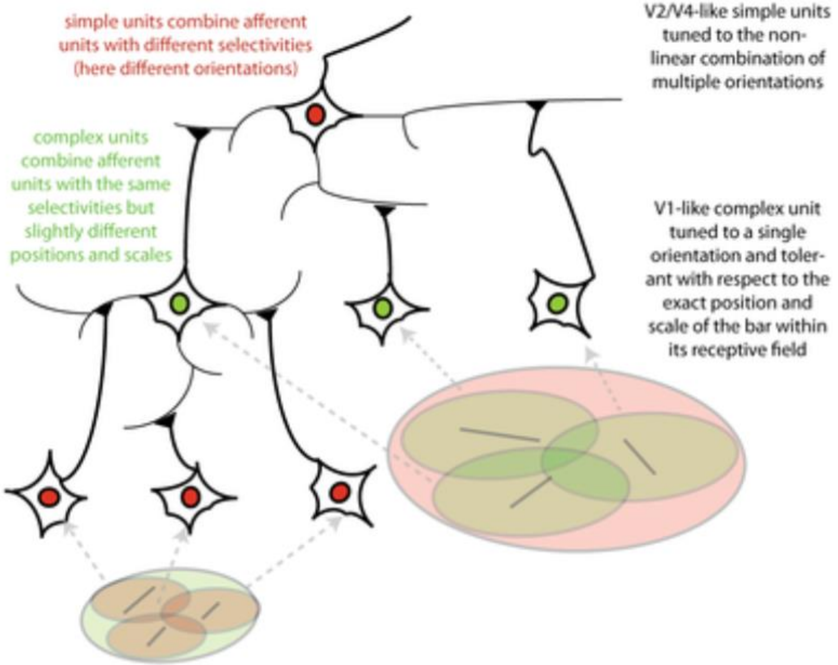
COGS 181, Fall 2017

Neural Networks and Deep Learning

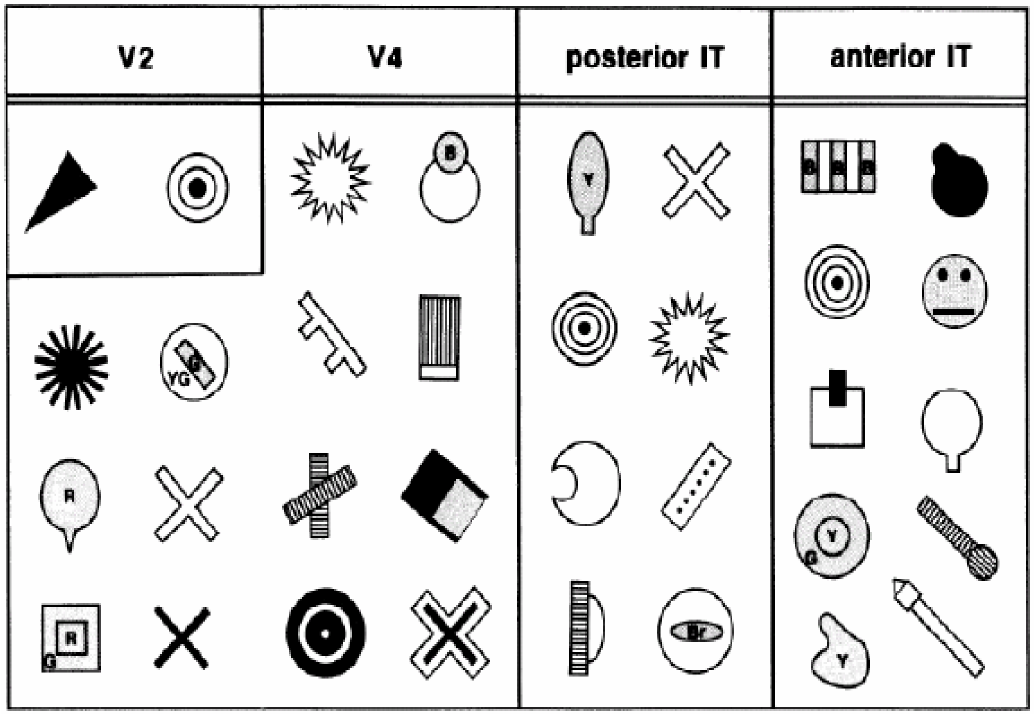
Lecture 1: Basics

Visual Representation



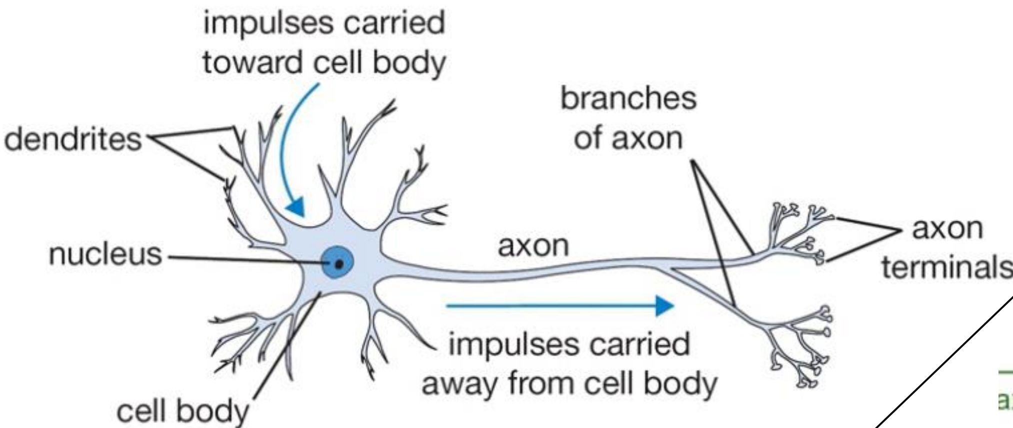


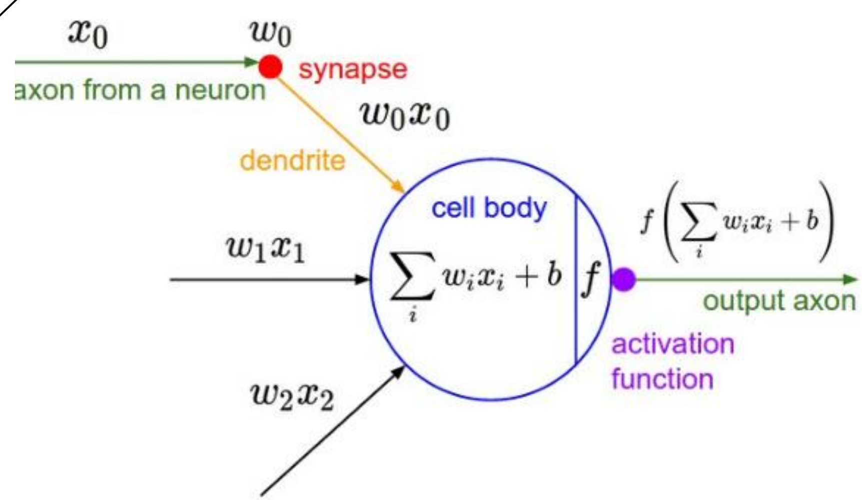
Hubel and Wiesel Model

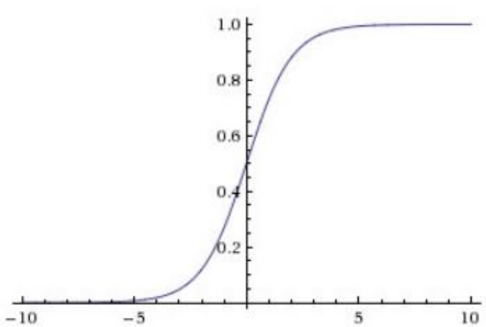


Kobatake and Tanaka, 1994

Perceptron

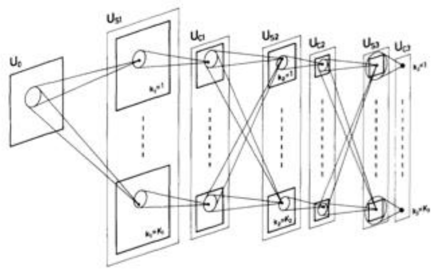




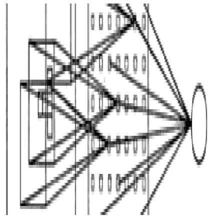


Sigmoid function
$f(x) = \frac{1}{1+e^{-x}}$

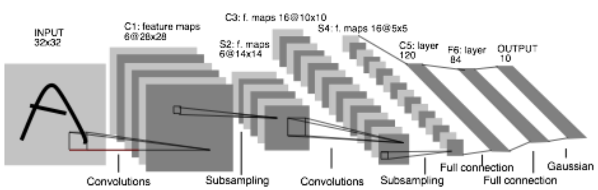
History of ConvNets



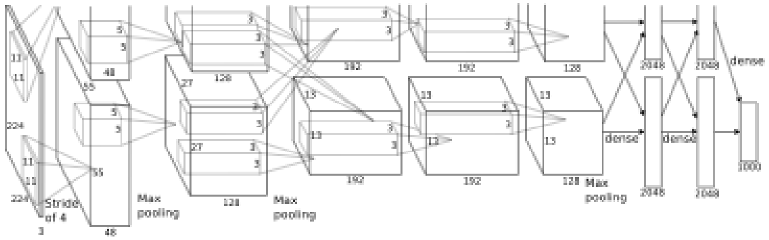
Fukushima 1980
Neocognitron



Rumelhart, Hinton, Williams 1986
“T” versus “C” problem



LeCun et al. 1989-1998
Hand-written digit reading

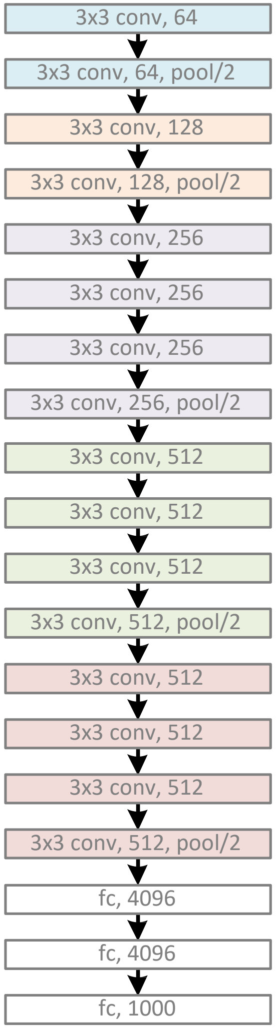


Krizhevksy, Sutskever, Hinton 2012
ImageNet classification

ImageNet experiments

VGG

(Karen Simonyan and Andrew Zisserman)



19 layers
(ILSVRC 2014)

The very deep ConvNets were the basis of our ImageNet ILSVRC-2014 submission, where our team (VGG) secured the first and the second places in the localisation and classification tasks respectively. After the competition, we further improved our models, which has lead to the following ImageNet classification results:

Model	top-5 classification error on ILSVRC-2012 (%)	
	validation set	test set
16-layer	7.5%	7.4%
19-layer	7.5%	7.3%
model fusion	7.1%	7.0%

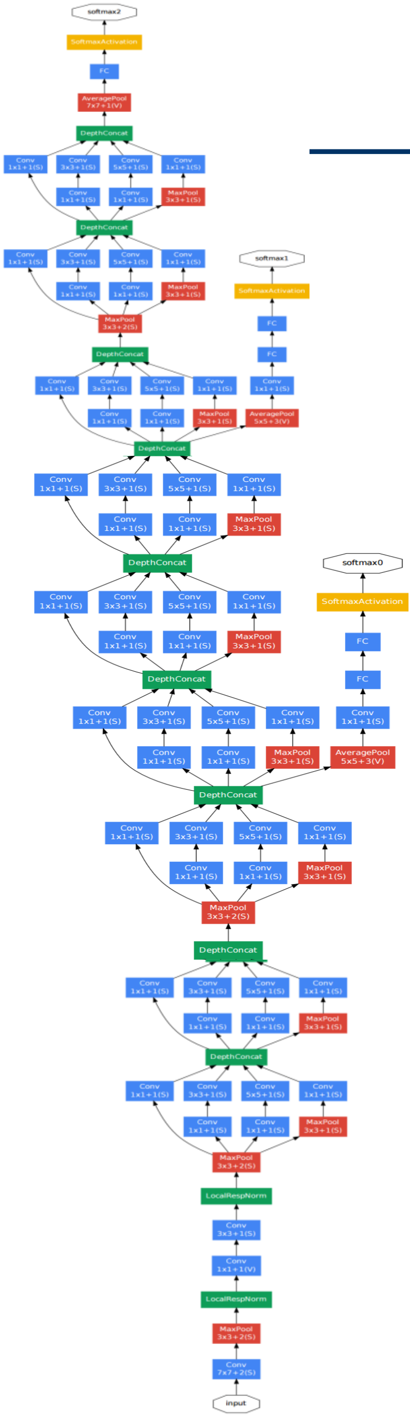
Very deep models generalize well to other datasets. A combination of multi-scale convolutional features and a linear SVM matches or outperforms more complex recognition pipelines built around less deep features. Our results on PASCAL VOC and Caltech image classification benchmarks are as follows:

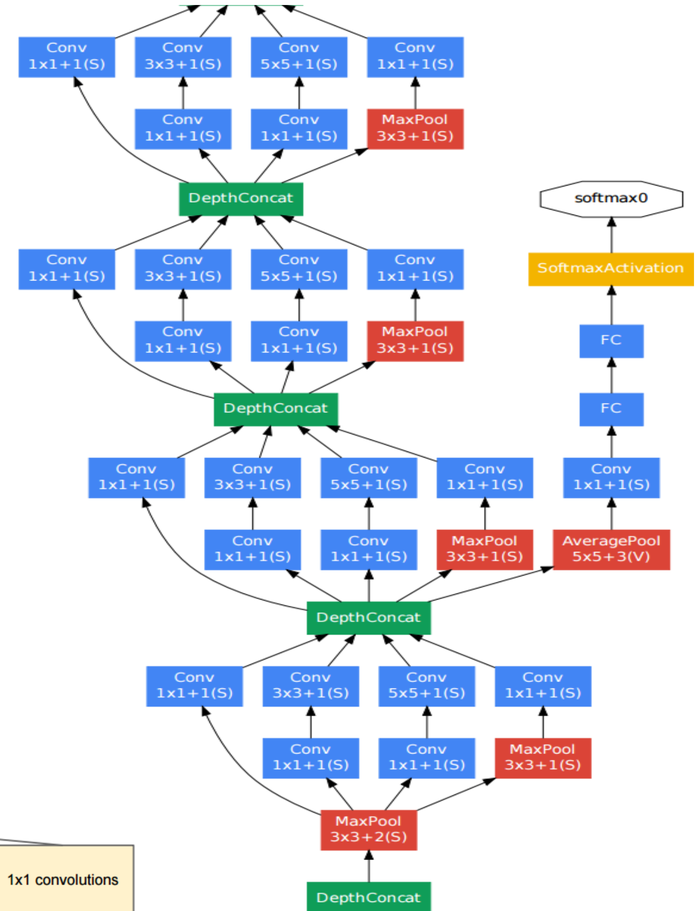
Model	VOC-2007 (mean AP, %)	VOC-2012 (mean AP, %)	Caltech-101 (mean class recall, %)	Caltech-256 (mean class recall, %)
16-layer	89.3	89.0	91.8±1.0	85.0±0.2
19-layer	89.3	89.0	92.3±0.5	85.1±0.3
model fusion	89.7	89.3	92.7±0.5	86.2±0.3

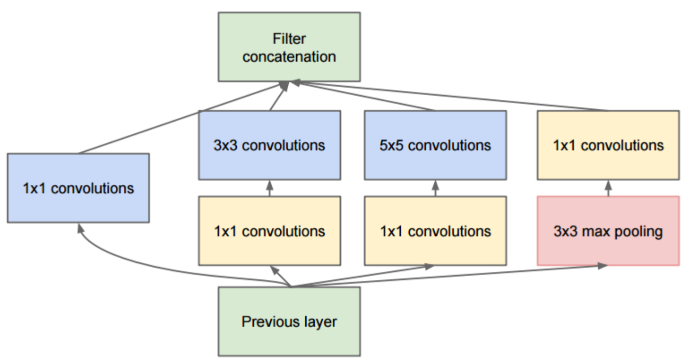
Widely adopted in computer vision as a pre-trained model for other tasks.

GoogLeNet

(Szegedy et al., 2014)

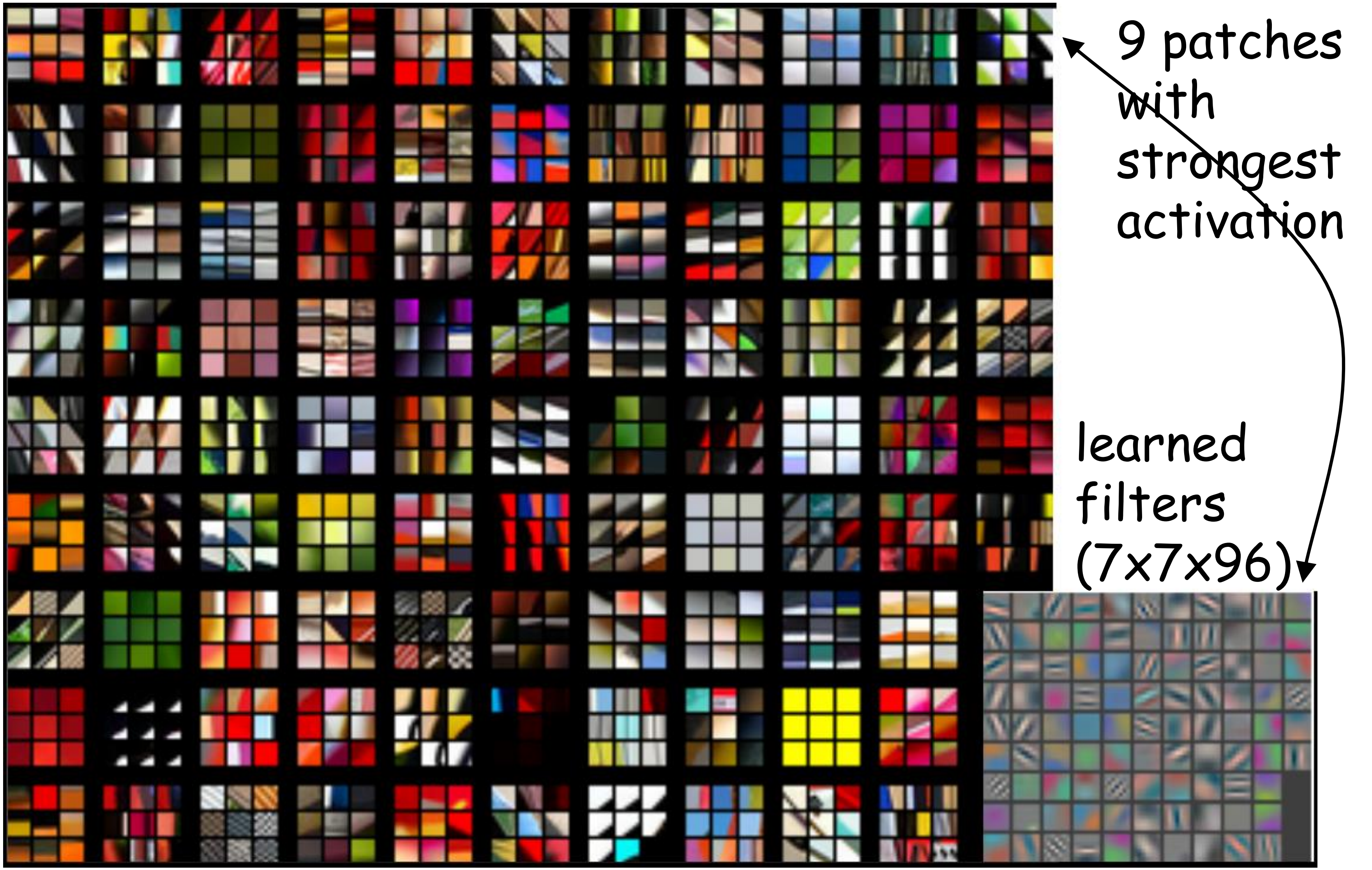






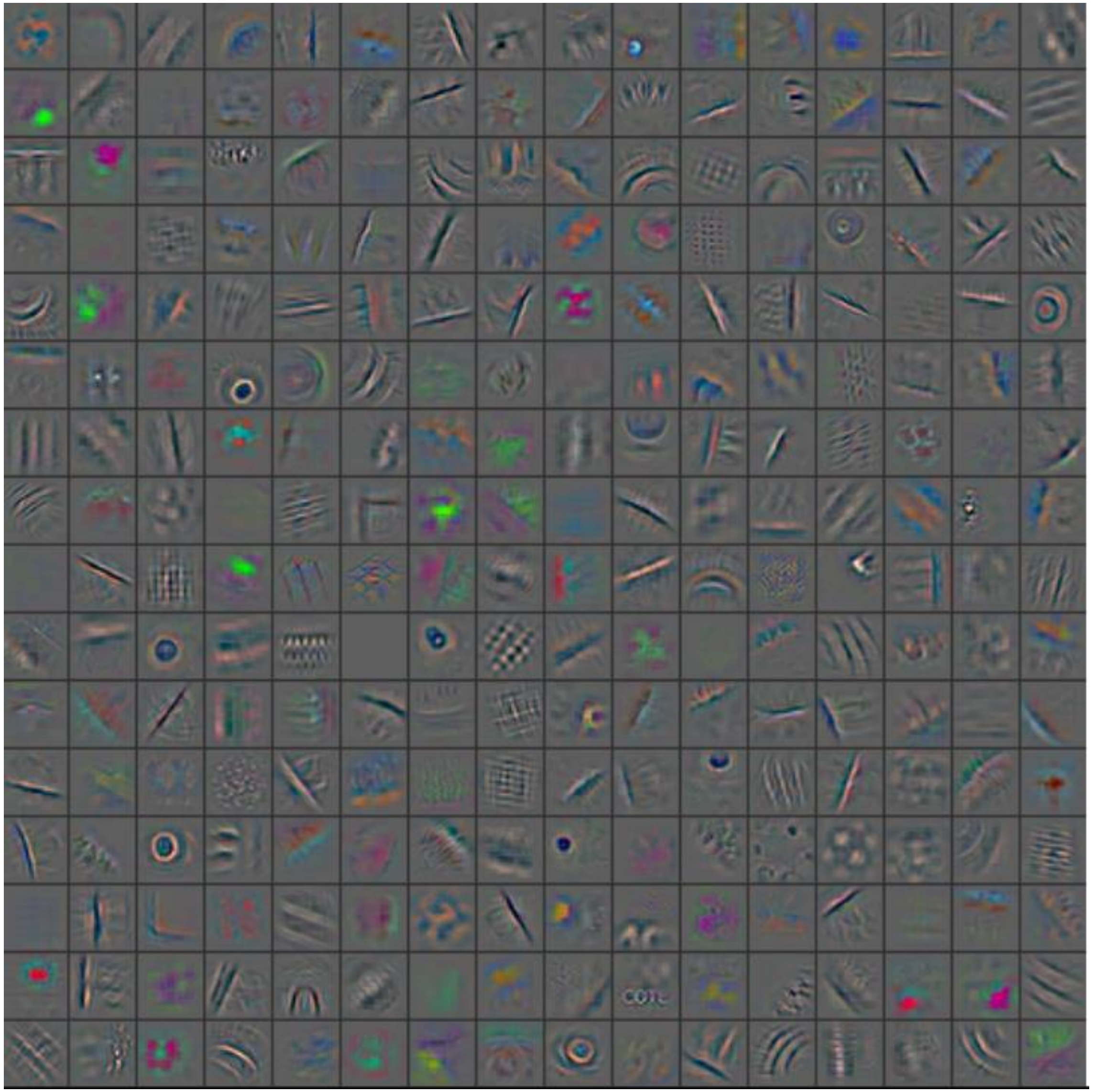
inception modules

Learned convolutional filters: Stage 1



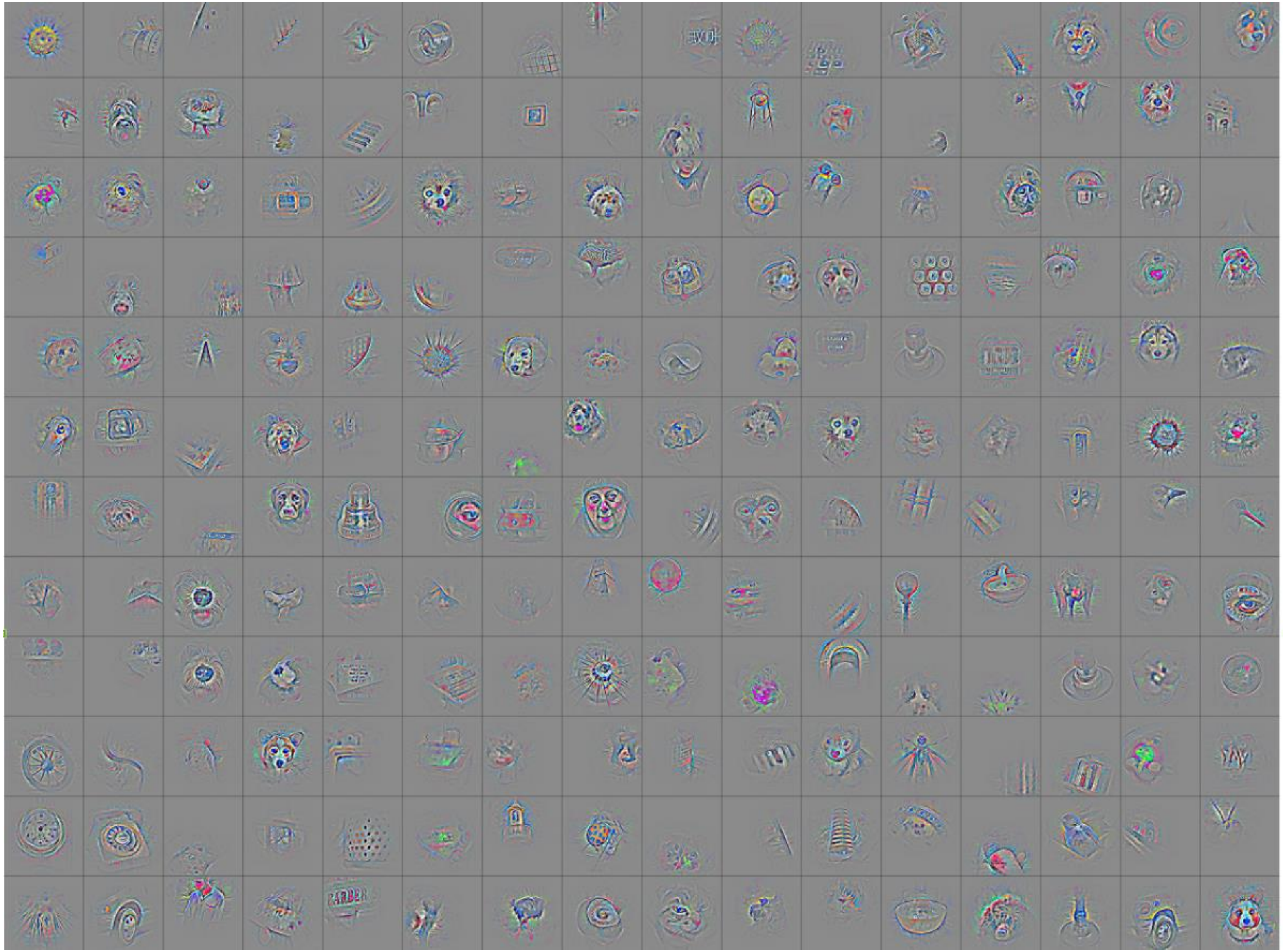
Visualizing and understanding convolutional neural networks. Zeiler, Matthew D., and Rob Fergus. *arXiv preprint arXiv:1311.2901* (2013).

Strongest activations: Stage 2



Visualizing and understanding convolutional neural networks. Zeiler, Matthew D., and Rob Fergus. *arXiv preprint arXiv:1311.2901* (2013).

Strongest activations: Stage 5



Visualizing and understanding convolutional neural networks. Zeiler, Matthew D., and Rob Fergus. *arXiv preprint arXiv:1311.2901* (2013).

Resources

http://deeplearning.net/software_links/

- **Theano** – CPU/GPU symbolic expression compiler in python (from MILA lab at University of Montreal)
- **Caffe** (Caffe2) -Caffe is a deep learning framework made with expression, speed, and modularity in mind.Caffe is a deep learning framework made with expression, speed, and modularity in mind.
- **Torch** (PyTorh)– provides a Matlab-like environment for state-of-the-art machine learning algorithms in lua
- **MxNET -** MxNET is fast, concise, distributed deep learning framework based on MShadow. It is a lightweight and easy extensible C++/CUDA neural network toolkit with friendly Python/Matlab interface for training and prediction.
- **TensorFlow-** Created and maintained by Google (Python, CPU/GPU). Include nearly all popular models and frameworks in deep learning.
- **MatConvNet-** Convolutional neural networks in matlab. Easy to use but not very popular in deep learning.
- **CNTK-** Developed and maintained by Microsoft; good for speech recognition and recurrent neural network.

CUDA + C++ + Python

Deep learning platforms

Deep learning software by name [edit]

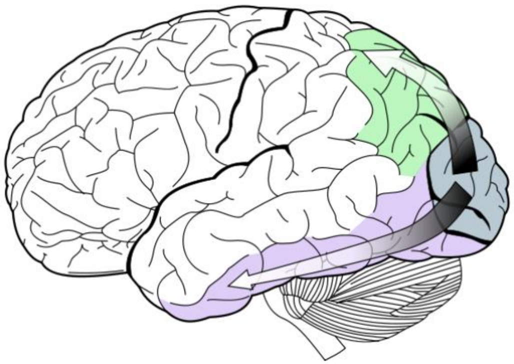
Software	Creator	Software license[a]	Open source	Platform	Written in	Interface	OpenMP support	OpenCL support	CUDA support	Automatic differentiation[1]	Has pretrained models	Recurrent nets	Convolutional nets	RBM/DBNs	Parallel execution (multi node)
Apache SINGA	Apache Incubator	Apache 2.0	Yes	Linux, Mac OS X, Windows	C++	Python, C++, Java	No	Yes	Yes	?	Yes	Yes	Yes	Yes	Yes
Caffe	Berkeley Vision and Learning Center	BSD license	Yes	Linux, Mac OS X, Windows[2]	C++	Python, MATLAB	Yes	Under development[3]	Yes	Yes	Yes[4]	Yes	Yes	No	?
Deeplearning4j	Skymind engineering team; Deeplearning4j community; originally Adam Gibson	Apache 2.0	Yes	Linux, Mac OS X, Windows, Android (Cross-platform)	java	Java, Scala, Clojure, Python (Keras)	Yes	On roadmap[5]	Yes[6]	Computational Graph	Yes[7]	Yes	Yes	Yes	Yes[8]
Dlib	Davis King	Boost Software License	Yes	Cross-Platform	C++	C++	Yes	No	Yes	Yes	Yes	No	Yes	Yes	Yes
Keras	François Chollet	MIT license	Yes	Linux, Mac OS X, Windows	Python	Python	Only if using Theano as backend	Under development for the Theano backend (and on roadmap for the TensorFlow backend)	Yes	Yes	Yes[9]	Yes	Yes	Yes	Yes[10]
MatConvNet	Andrea Vedaldi,Karel Lenc	BSD license	Yes	Windows, Linux[11] (OSX via Docker on roadmap)	C++	MATLAB, C++,	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes
Microsoft Cognitive Toolkit	Microsoft Research	MIT license[12]	Yes	Windows, Linux[13] (OSX via Docker on roadmap)	C++	Python, C++, Command line,[14] BrainScript[15] (.NET on roadmap[16])	Yes[17]	No	Yes	Yes	Yes[18]	Yes[19]	Yes[19]	No[20]	Yes[21]
MXNet	Distributed (Deep) Machine Learning Community	Apache 2.0	Yes	Linux, Mac OS X, Windows,[22][23] AWS, Android,[24] iOS, JavaScript[25]	Small C++ core library	C++, Python, Julia, Matlab, JavaScript, Go, R, Scala, Perl	Yes	On roadmap[26]	Yes	Yes[27]	Yes[28]	Yes	Yes	Yes	Yes[29]
Neural Designer	Artelnics	Proprietary	No	Linux, Mac OS X, Windows	C++	Graphical user interface	Yes	No	No	?	?	No	No	No	?
OpenNN	Artelnics	GNU LGPL	Yes	Cross-platform	C++	C++	Yes	No	No	?	?	No	No	No	?
TensorFlow	Google Brain team	Apache 2.0	Yes	Linux, Mac OS X, Windows[30]	C++, Python	Python (Keras), C/C++, Java, Go, R[31]	No	On roadmap[32][33]	Yes	Yes[34]	Yes[35]	Yes	Yes	Yes	Yes
Theano	Université de Montréal	BSD license	Yes	Cross-platform	Python	Python	Yes	Under development[36]	Yes	Yes[37][38]	Through Lasagne's model zoo[39]	Yes	Yes	Yes	Yes[40]
Torch	Ronan Collobert, Koray Kavukcuoglu, Clement Farabet	BSD license	Yes	Linux, Mac OS X, Windows,[41] Android,[42] iOS	C, Lua	Lua, LuaJIT,[43] C, utility library for C++/OpenCL[44]	Yes	Third party implementations[45][46]	Yes[47][48]	Through Twitter's Autograd[49]	Yes[50]	Yes	Yes	Yes	Yes[51]
Wolfram Mathematica	Wolfram Research	Proprietary	No	Windows, Mac OS X, Linux, Cloud computing	C++	Wolfram Language	No	No	Yes	Yes	Yes[52]	Yes	Yes	Yes	Yes

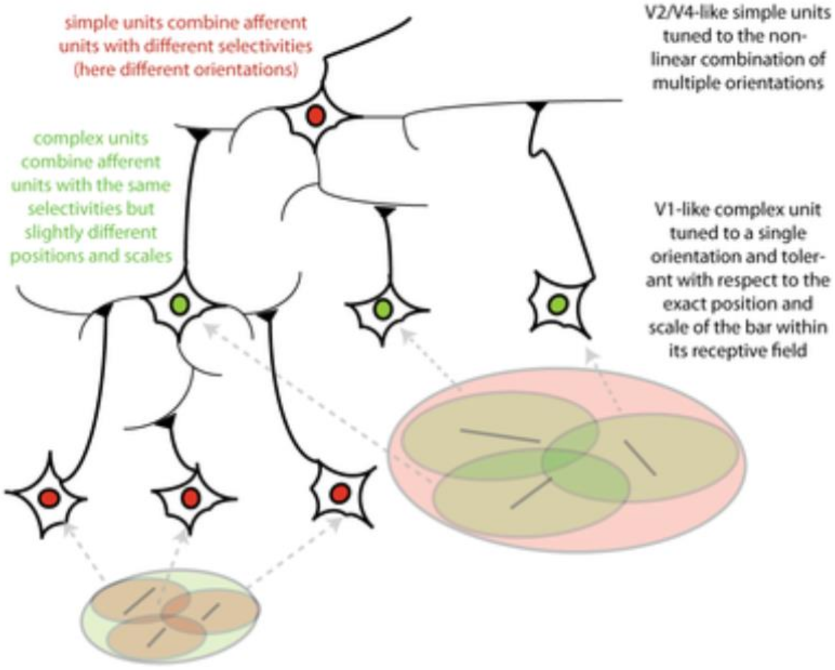
https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software

Also: https://deeplearning4j.org/compare-dl4j-torch7-pylearn

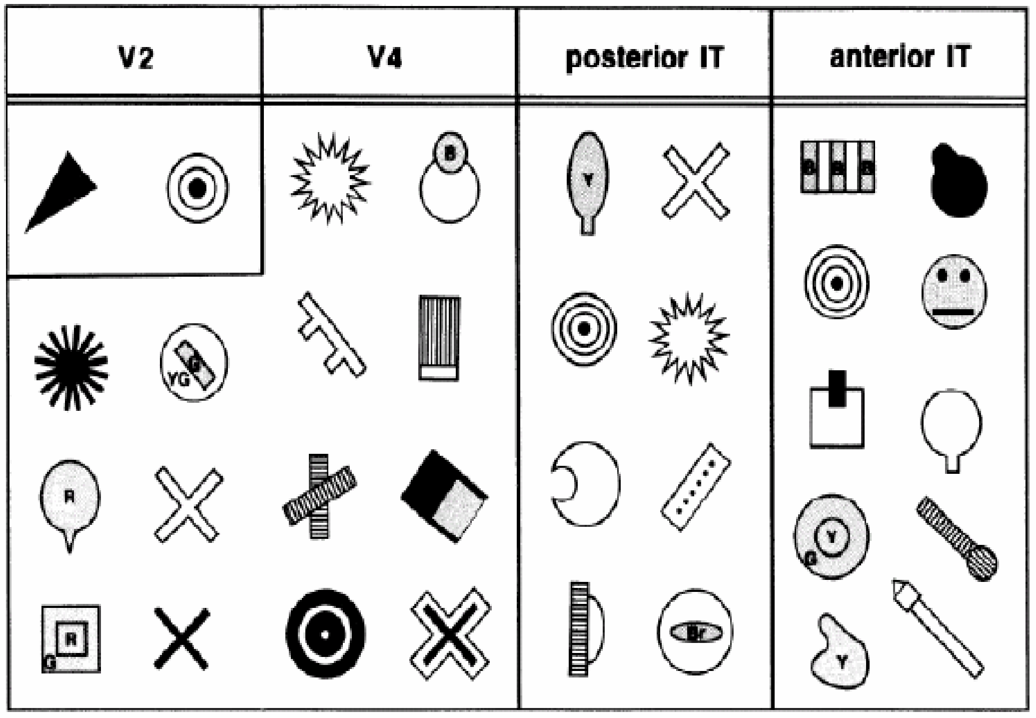
Visual Representation

(1) Hierarchical
(2) Compositional





Hubel and Wiesel Model



Kobatake and Tanaka, 1994

Some notations that we will be using

$S = \{\mathbf{x}_i, i = 1..n\}$: A set S with n samples. i goes from 1 to n.

Important to note: the order in a set doesn't matter, e.g.

$\{a, b, c\} = \{c, a, b\}$

$\{a, b, c\} \neq \{c, a, b, d\}$

$\mathbf{x}_i = (x_{i1}, ..., x_{im})$: A row vector of m elements. $\mathbf{x}_i = (22, 1, 0, 160, 180)$

Important to note: the order in a vector matters, e.g.

$(a, b, c) \neq (d, b, c)$

Sometimes, we also use $\mathbf{x}_i =< x_{i1}, ..., x_{im} >$

In matlab, $\mathbf{x}_i = [x_{i1} \;\; x_{i2} \;\; x_{i3}]$

Several Pairs of Concepts

$$y : class\ label \qquad x : data(features)$$

Generative

$$p(x, y)$$

Speaking Spanish
Speaking French

Discriminative

$$p(y|x)$$

Classify if someone is speaking
Spanish or French

Parametric

$$y = f(x)$$

Flooding?
weather + month + location

Non-parametric

$$y = \sum_{k=1}^{K} \alpha_k f_k(x)$$

Flooding?
Every 10/03 in the history.

Supervised vs.

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

Unsupervised

$$S_{training} = \{(\mathbf{x}_i), i = 1..n\}$$

Basic concepts (supervised)

Training (supervised)

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im}), x \in \mathcal{R}, \quad \mathbf{x} \in \mathcal{R}^k$$

blood presure	age	male or female	weight (lb)	height (cm)
y_1=131	$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
y_2=150	$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
y_3=105	$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

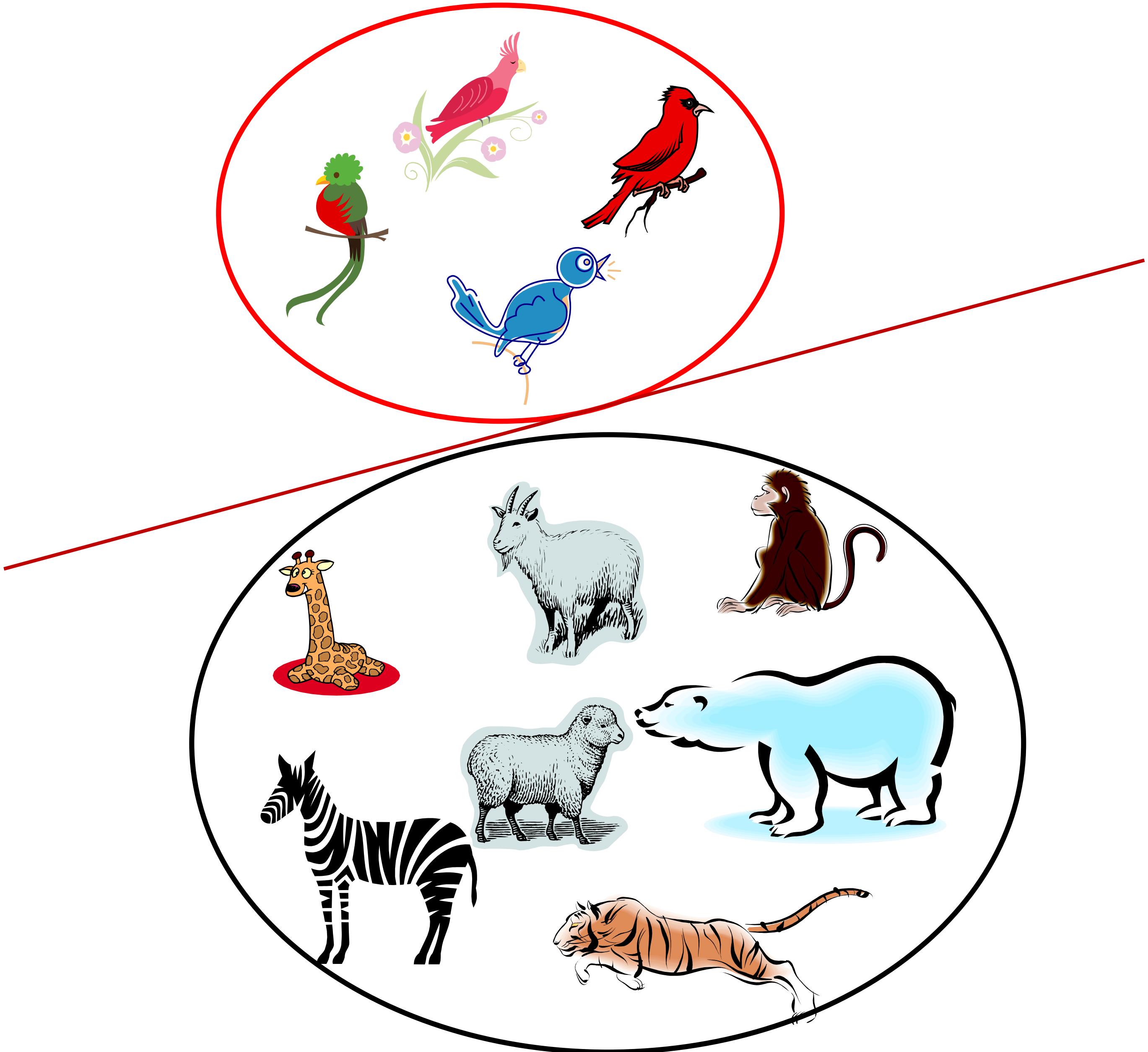
In supervised setting during training, y_i (the solution) to each sample x_i is provided.

Testing:

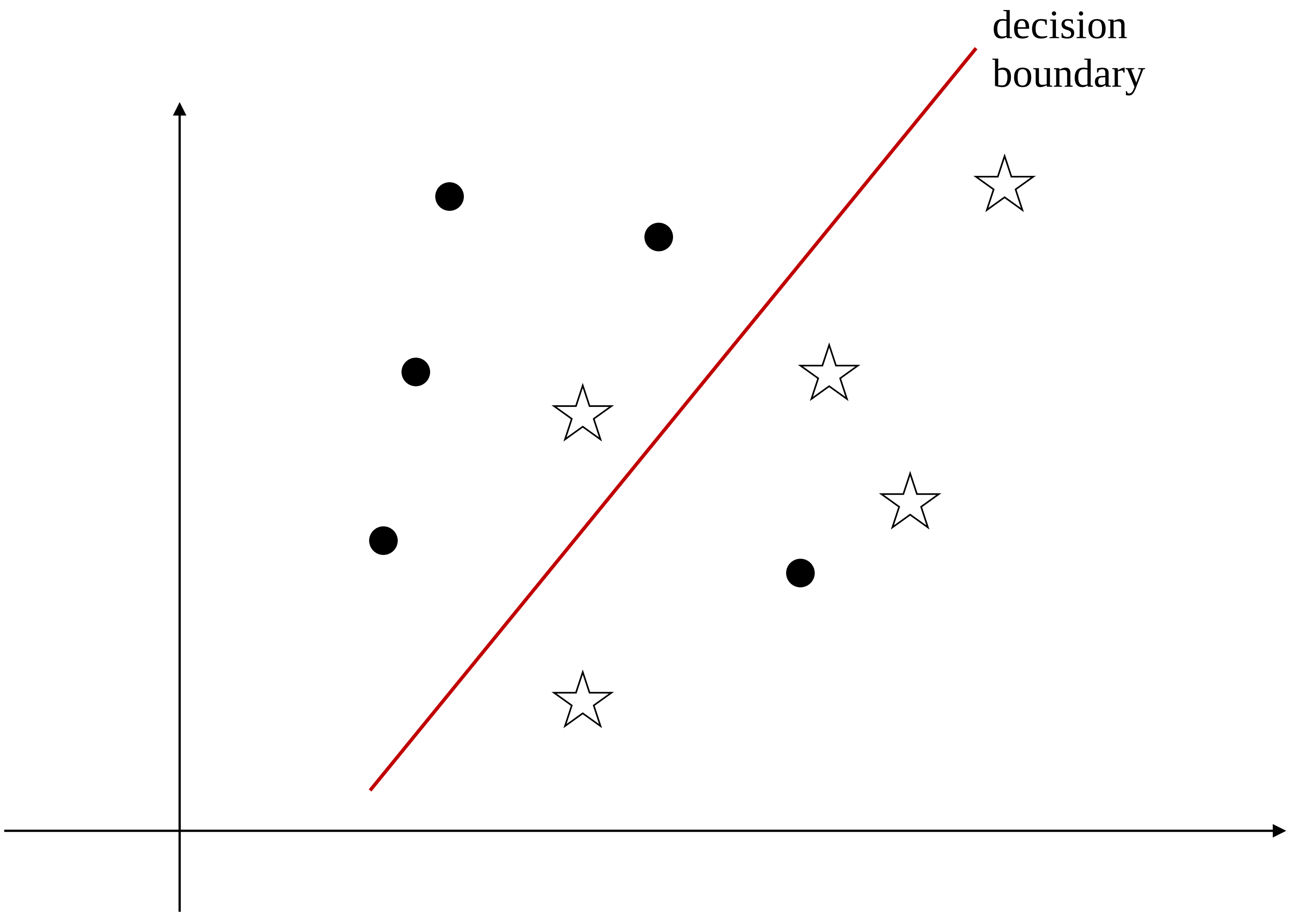
$$S_{testing} = \{(\mathbf{x}_i), i = 1..u\}, what\ is\ y_i?$$

blood presure	age	male or female	weight (lb)	height (cm)
y_1=?	$x_{11} = 32$	$x_{12} = F$	$x_{13} = 130$	$x_{14} = 180$
y_2=?	$x_{21} = 11$	$x_{22} = M$	$x_{23} = 52$	$x_{24} = 135$

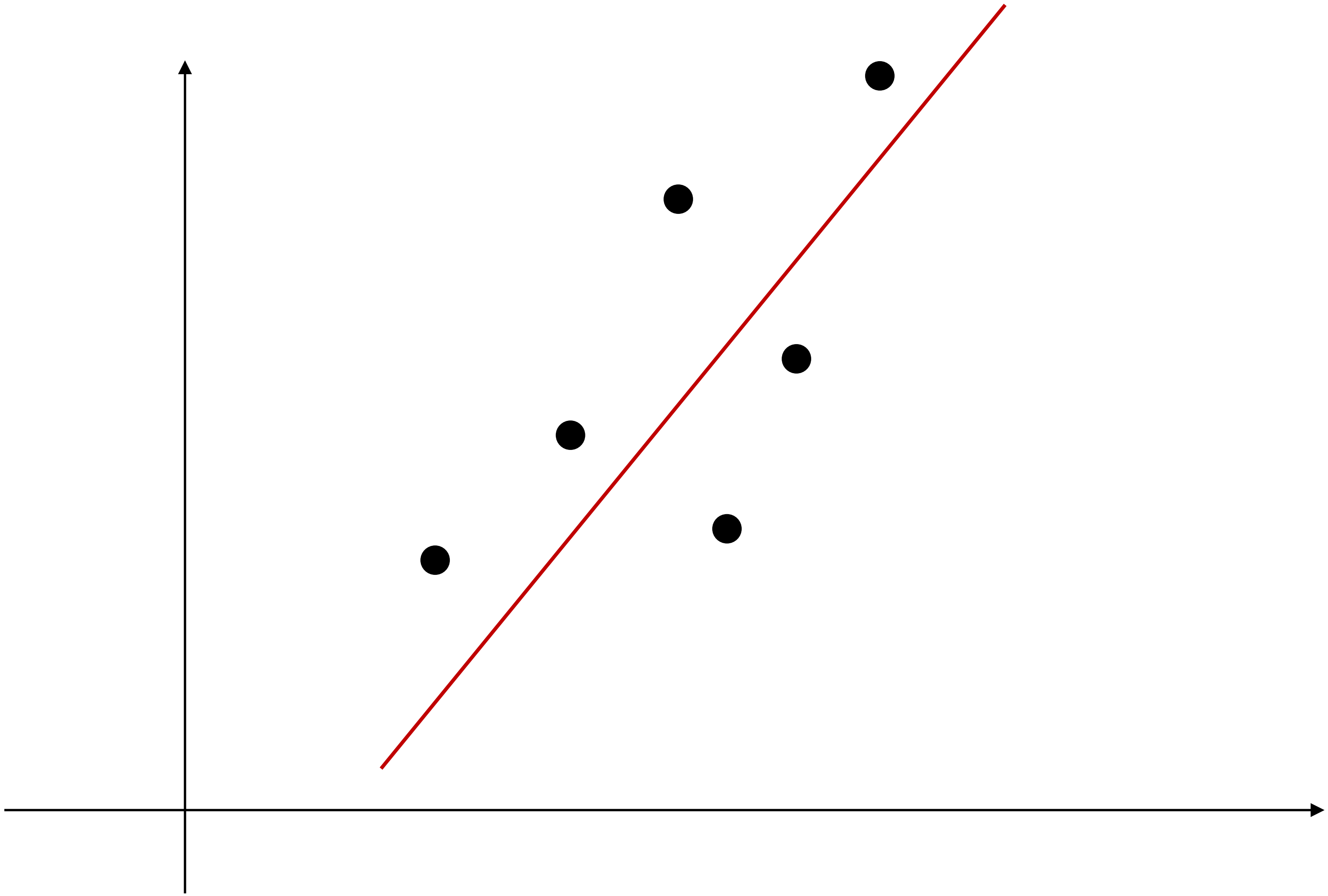
Supervised



Classification (supervised)



Regression (supervised)



Basic concepts (unsupervised)

Training (unsupervised)

$$S_{training} = \{(\mathbf{x}_i), i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im})$$

age	male or female	weight (lb)	height (cm)
$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

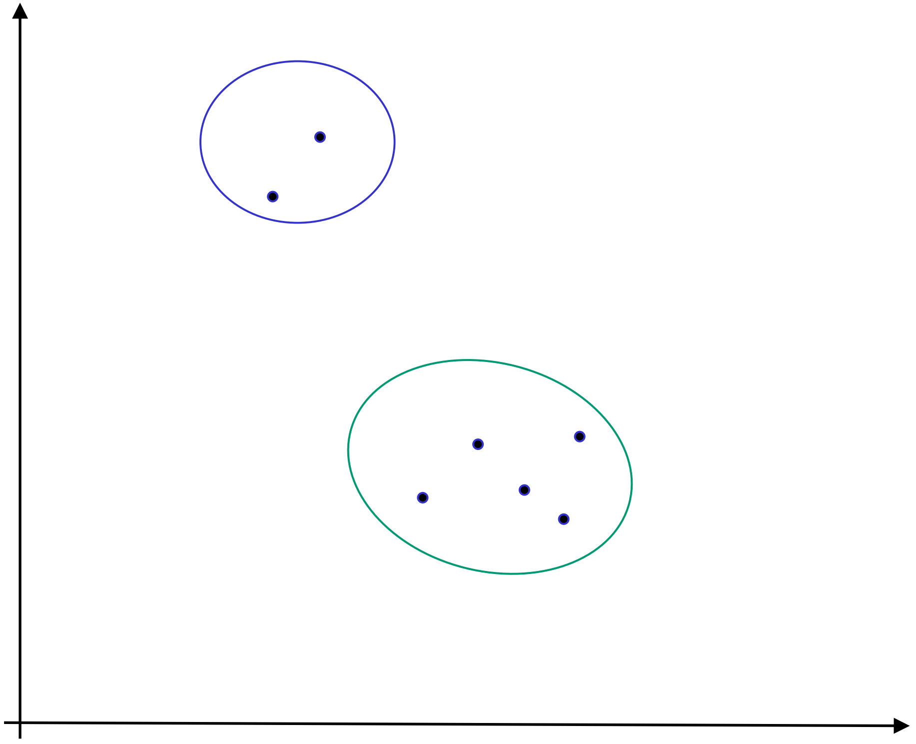
In unsupervised setting during training, no answer is provided to each sample x_i

Testing:

$$S_{testing} = \{(\mathbf{x}_i), i = 1..u\}$$

what is the chance	age	male or female	weight (lb)	height (cm)
$p(\mathbf{x}_1)$?	$x_{11} = 32$	$x_{12} = F$	$x_{13} = 130$	$x_{14} = 180$
$p(\mathbf{x}_2)$?	$x_{21} = 11$	$x_{22} = M$	$x_{23} = 52$	$x_{24} = 135$

Supervised vs. unsupervised



Features

What defines features?

How are they given?

What is the difference between features and the input?

How to compute them?

How to evaluate them?

How to use them?

Features

For the moment, we assume that features are given as data input:

$$S = \{(\mathbf{x}_i), i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im})$$

age	male or female	weight (lb)	height (cm)
$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

Mathematical representation for features

$$S = \{(\mathbf{x}_i), i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im})$$

age	male or female	weight (lb)	height (cm)
$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

Discrete variable: age $x_{i1} \in N^+$

You can use a number for the representation.

Continues variable: weight $x_{i3} \in R^+$

You can use a positive real value for the representation.

Mathematical representation for features

$$S = \{(\mathbf{x}_i), i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im})$$

age	male or female	weight (lb)	height (cm)
$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

Gender variable: $x_{i2} \in \{Male, Female\}$?

$x_{i2} = 0$, if Male

$x_{i2} = 1$, if Female

Mathematical representation for features

$$S = \{(\mathbf{x}_i), i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im})$$

What if it is a city: $x_{i2} \in \{LosAngeles, SanDiego, Irvine\}$?

We ue a coding strategy by expanding the features.
For N number of possible states, we expand the features into N-dimenstional.

One-hot encoding:

	coded values
Los Angeles	1, 0, 0
San Diego	0, 1, 0
Irvine	0, 0, 1

Pros: we can naturally deal with any type of input (can associate confidence directly).

Cons: the feature dimension has become much larger.

Input matrix

$$S = \{\mathbf{x}_i, i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im})$$

age	male or female	weight (lb)	height (cm)
$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

If we write each sample as a row vector:

$$\mathbf{x}_1 = (22, 1, 0, 160, 180)$$

$$X = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} \qquad X \in R^{n \times m}$$

$$X = \begin{pmatrix} 22 & 1 & 0 & 160 & 180 \\ 51 & 1 & 0 & 190 & 175 \\ 43 & 0 & 1 & 120 & 165 \end{pmatrix}$$

Input matrix

$$S = \{\mathbf{x}_i, i = 1..n\} \qquad \mathbf{x}_i = (x_{i1}, ..., x_{im})^T$$

age	male or female	weight (lb)	height (cm)
$x_{11} = 22$	$x_{12} = M$	$x_{13} = 160$	$x_{14} = 180$
$x_{21} = 51$	$x_{22} = M$	$x_{23} = 190$	$x_{24} = 175$
$x_{31} = 43$	$x_{32} = F$	$x_{33} = 120$	$x_{34} = 165$

More often we write each sample as a COLUMN vector:

$$\mathbf{x}_1 = \begin{pmatrix} 22 \\ 1 \\ 0 \\ 160 \\ 180 \end{pmatrix}$$

$$X = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) \qquad X \in R^{m \times n}$$

$$X = \begin{pmatrix} 22 & 51 & 43 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 160 & 190 & 120 \\ 180 & 175 & 165 \end{pmatrix}$$

Importance of features

136	136	132	129	129	125	122	122	117	114	114	114	112	112	110	109	109
135	135	134	133	133	131	126	126	120	115	115	110	108	108	105	103	103
133	133	133	134	134	133	129	129	123	119	119	111	108	108	105	101	101
133	133	133	134	134	133	129	129	123	119	119	111	108	108	105	101	101
133	133	133	133	133	133	131	131	128	125	125	118	115	115	111	107	107
136	136	133	132	132	132	133	133	132	131	131	126	123	123	118	114	114
136	136	133	132	132	132	133	133	132	131	131	126	123	123	118	114	114
136	136	136	136	136	135	132	132	129	127	127	128	125	125	122	119	119
136	136	137	136	136	135	133	133	130	129	129	129	127	127	124	122	122
134	134	135	135	135	135	133	133	132	131	131	129	127	127	125	124	124
134	134	135	135	135	135	133	133	132	131	131	129	127	127	125	124	124
129	129	130	131	131	132	132	132	131	131	131	128	127	127	125	124	124
122	122	123	125	125	126	127	127	128	128	128	127	126	126	125	123	123
122	122	123	125	125	126	127	127	128	128	128	127	126	126	125	123	123
119	119	119	119	119	120	120	120	120	121	121	123	121	121	121	121	121
120	120	118	116	116	114	113	113	112	112	112	114	112	112	113	114	114
120	120	118	116	116	114	113	113	112	112	112	114	112	112	113	114	114
122	122	119	115	115	112	109	109	107	106	106	104	104	104	104	106	106
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	98	96	95
128	128	126	122	122	116	112	112	108	107	107	100	98	98	93	92	92
[illegible]	[illegible]	[illegible]	[illegible]	122	116	112	112	108	107	107	100	98	98	93	92	92
130	130	127	124	124	120	117	117	114	113	113	105	102	102	99	96	96
131	131	128	125	125	122	119	119	117	116	116	111	109	109	106	103	103
131	131	128	125	125	122	119	119	117	116	116	111	109	109	106	103	103
129	129	128	125	125	122	120	120	118	116	116	111	109	109	107	105	105
124	124	124	123	123	122	119	119	117	115	115	112	109	109	109	107	107
116	116	119	120	120	120	119	119	116	113	113	113	113	113	111	109	109
116	116	119	120	120	120	119	119	116	113	113	113	113	113	111	109	109
110	110	115	118	118	120	119	119	116	115	115	113	110	110	109	107	107
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
106	106	106	105	105	107	109	109	111	112	112	110	109	109	104	101	101
[illegible]	[illegible]	[illegible]	[illegible]	99	98	99	99	101	103	103	106	104	104	102	100	100

Columns 1 through 27

136 135 133 133 133 136 136 136 136 134 134 129 122 122 119 120 120 122 128 128 128 130 131 131 129 124 116

Columns 28 through 54

116 110 106 106 106 108 108 114 115 114 114 114 115 115 108 104 104 102 103 103 103 103 100 100 98 93 101

Columns 55 through 81

101 107 100 100 82 61 61 47 44 49 49 50 51 51 53 55 55 58 59 60 60 63 64 64 65 66 64

Columns 82 through 108

64 64 62 62 60 62 62 59 55 55 55 60 60 60 58 59 59 62 64 68 68 67 54 54 43 47 59

Columns 109 through 135

59 69 70 70 72 73 73 75 76 79 79 80 70 70 55 83 149 149 169 139 139 135 151 151 170 171 171

Columns 136 through 162

171 165 136 136 114 139 139 165 161 164 164 174 179 179 168 157 156 156 158 165 165 154 172 172 188 179 176

Columns 163 through 189

176 181 177 177 181 186 186 189 188 187 187 189 190 190 189 188 191 191 193 188 188 183 177 177 142 93 80

Columns 190 through 216

80 84 89 89 97 109 109 136 135 133 133 133 136 136 136 136 134 134 129 122 122 119 120 120 122 128 128

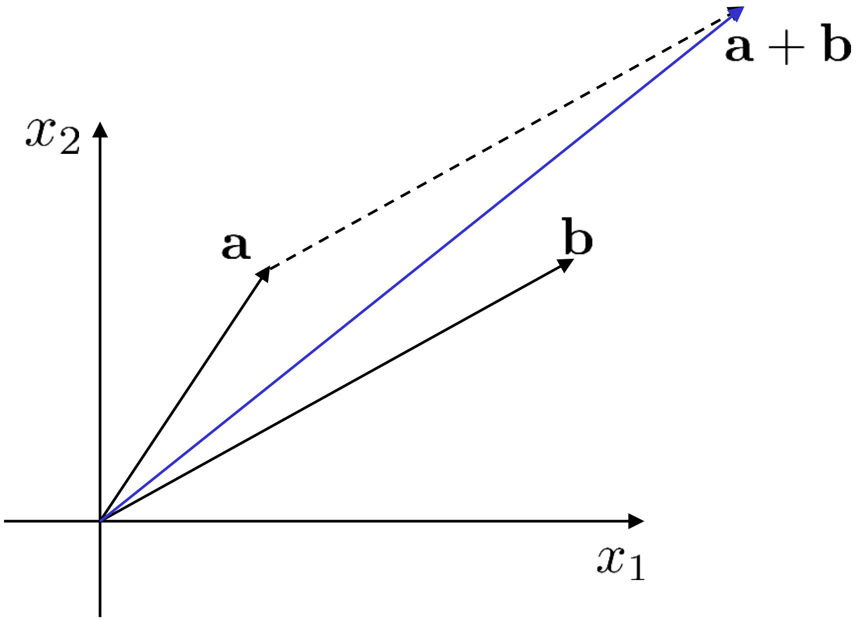
Columns 217 through 243

128 130 131 131 129 124 116 116 110 106 106 106 108 108 114 115 114 114 114 115 115 108 104 104 102 103 103

Columns 244 through 270

Vector

Addition:



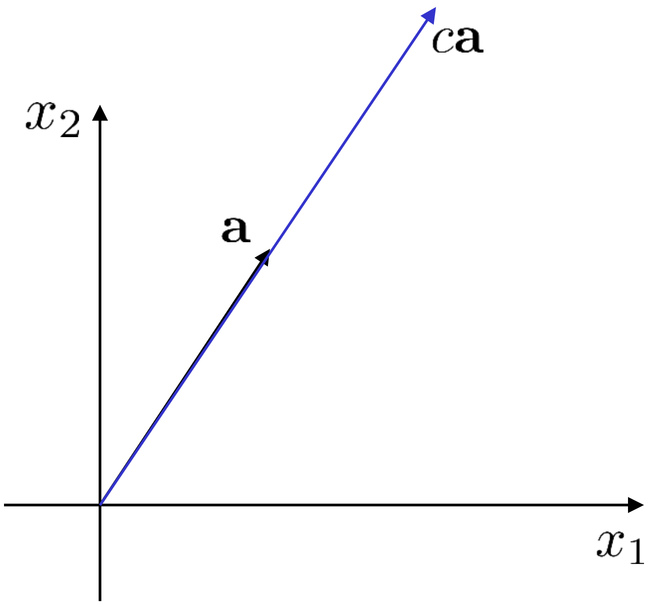
$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$\mathbf{a}+\mathbf{b} = \begin{pmatrix} a_1+b_1 \\ a_2+b_2 \\ a_3+b_3 \end{pmatrix}$$

It's still a vector in the same space as a and b.

Vector

Scaling:



$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

$$c \in R$$

$$c\mathbf{a} = \begin{pmatrix} c \times a_1 \\ c \times a_2 \\ c \times a_3 \end{pmatrix}$$

It's still a vector in the same space as a.

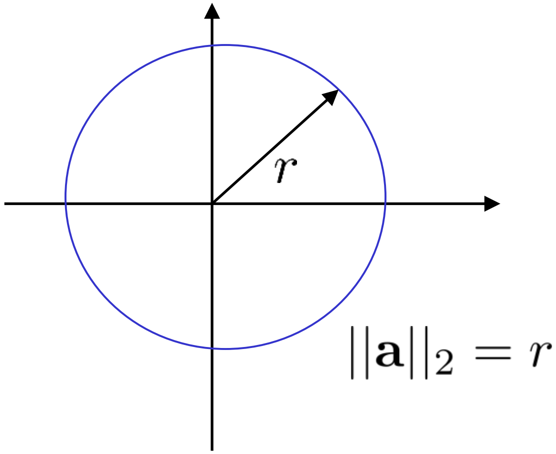
Norm

$$\mathbf{a} = (a_1, a_2, ..., a_n),\ a_i \in R$$

L2 Norm:

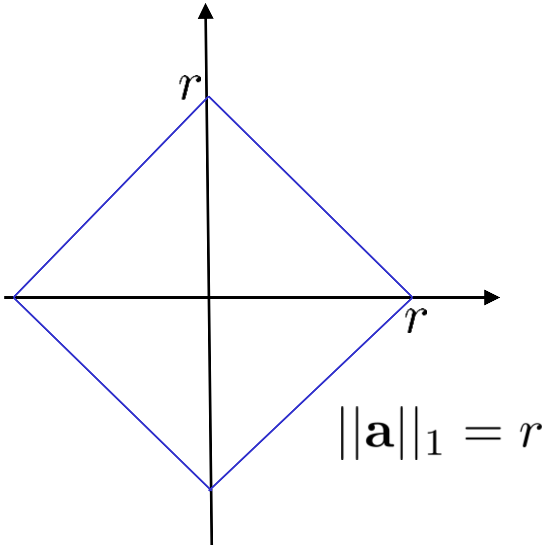
$$||\mathbf{a}||_2 = \sqrt{\sum_{i=1}^{n} a_i^2}$$

$$||\mathbf{a}||^2 = \sum_{i=1}^{n} a_i^2$$



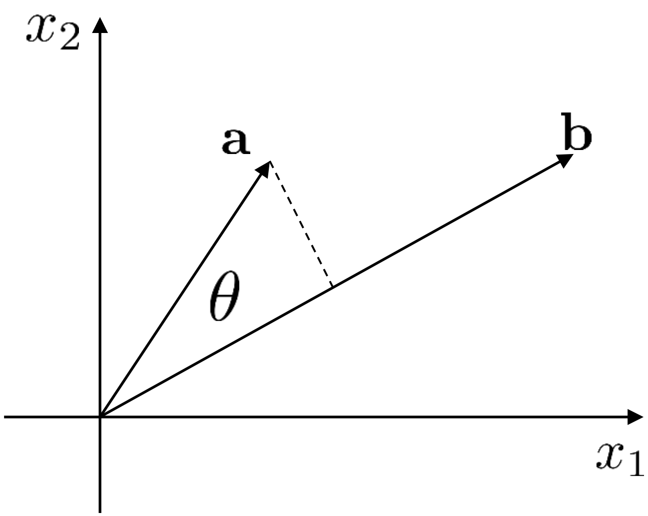
L1 Norm:

$$||\mathbf{a}||_1 = \sum_{i=1}^{n} |a_i|$$



Vector: Projection (inner product)

(one of the most important concepts in machine learning)



$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$< \mathbf{a}, \mathbf{b} > \ \equiv \mathbf{a} \cdot \mathbf{b} \ \equiv \mathbf{a}^T \mathbf{b} \ \equiv a_1 b_1 + a_2 b_2 + a_3 b_3$$

It's a scalar!

$$cos(\theta) = \frac{<\mathbf{a},\mathbf{b}>}{||\mathbf{a}||_2 \times ||\mathbf{b}||_2}$$