Deep Learning in Computer Vision

王鹏

peng.wang@nwpu.edu.cn

Pre-requisite

- Elementary programming ability
  - Most of assignments may be written in python
  - Suggest Pytorch as your deep learning platform
- Matrix/Vector Calculus, Linear Algebra, Probability and Statistics

Books and References

- Dive into Deep Learning, Aston Zhang and Zachary C. Lipton and Mu Li and Alexander J. Smola, http://www.d2l.ai, 2020.
- Deep Learning, Ian Goodfellow and Yoshua Bengio and Aaron Courville, MIT Press, 2016
- Pattern Recognition and Machine Learning, Christopher Bishop, Springer, 2011
- The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition, Springer, 2nd edition, 2016
- Convex Optimization, Stephen Boyd and Lieven Vandenberghe, Cambridge University Press, 2004

Grading Policy

- Three assignments: 15% x 3 = 45%
- Final exam: 45%
- In-class Participation: 10%

What we will learn

- Basic concepts about artificial intelligence, machine learning, computer vision, numeric optimization
- Deep learning: Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Attention
- The application of DL to CV tasks, such as image classification, object detection, image segmentation, image caption and visual question answering

What we will miss

- We focus on data-driven AI methods, so will ignore symbolic knowledge-driven methods
- We focus on discriminative ML algorithms, so will ignore generative ML algorithms such as graphic models
- We focus on supervised DL methods, so will ignore unsupervised and semi-supervised DL methods such as autoencoders
- We focus on mid-to-high level CV tasks, so will ignore low level tasks such as image denoise

Schedule and Syllabus

L1	Introduction	L11	DNNs for Pixel Prediction Tasks
L2	Linear Algebra Basics + Matrix Calculus	L12	DNNs for Object Detection (Two-stage, One-stage, Anchor-free)
L3	Optimization Basics	L13	DNNs for Object Tracking
L4	Machine Learning Basics	L14	DNNs for Vision-Language Tasks (CNN-LSTM, BERT, MCB)
L5	Neural Networks (Multi-layer Perceptrons, Activation Functions, Back-propagation)	L15	DNNs for Human-related Vision Tasks (Face Recognition, Pose Estimation and Person Re-identification)
L6	Training Neural Networks (SGD, Tricks for minimizing training errors and generalization gaps)	L16	DNNs for 3D Vision and DNNs for Action&Video
L7	Convolutional Neural Networks and Image Classification	L17	Graph Neural Networks for Visual Reasoning (GCN, GGNN, GAN)
L8	Recurrent Neural Networks	L18	AutoML for Vision Tasks (Classification, Segmentation, Detection)
L9	Attention Mechanisms (soft, hard, spatial, channel and self attentions)	L19	Visualization and Interpretability
L10	Generative Adversarial Networks	L20	Future Directions and Discussion

Computer Vision

Tasks

↓

Datasets

↓

Feature Engineering

↓

Machine Learning Algorithms

Image Deblur

Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms

Image Denoise

Computer Vision

Tasks

↓

Datasets

↓

Feature Engineering

↓

Machine Learning Algorithms

Super-resolution

Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms

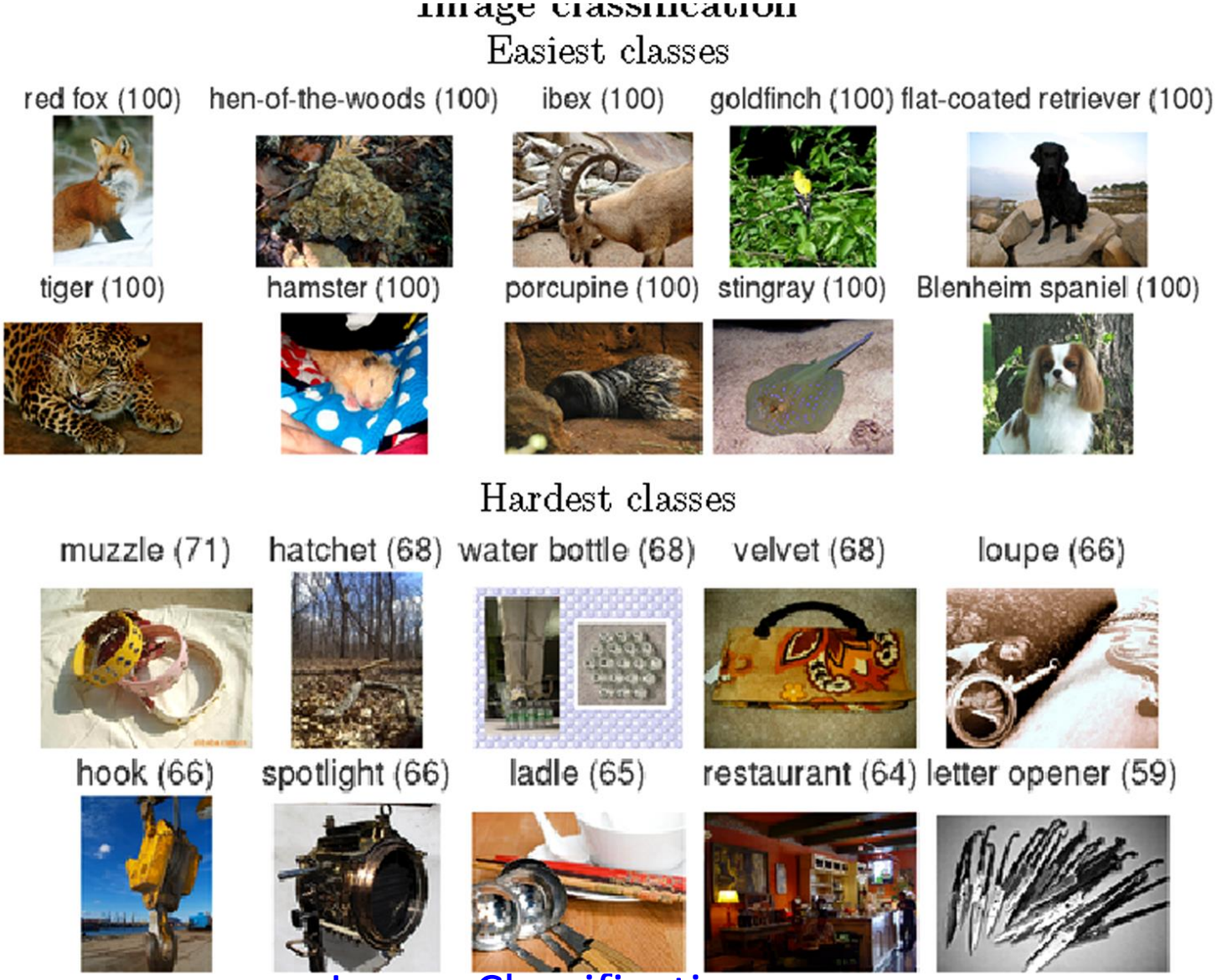


Image Classification

Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms

Image Segmentation

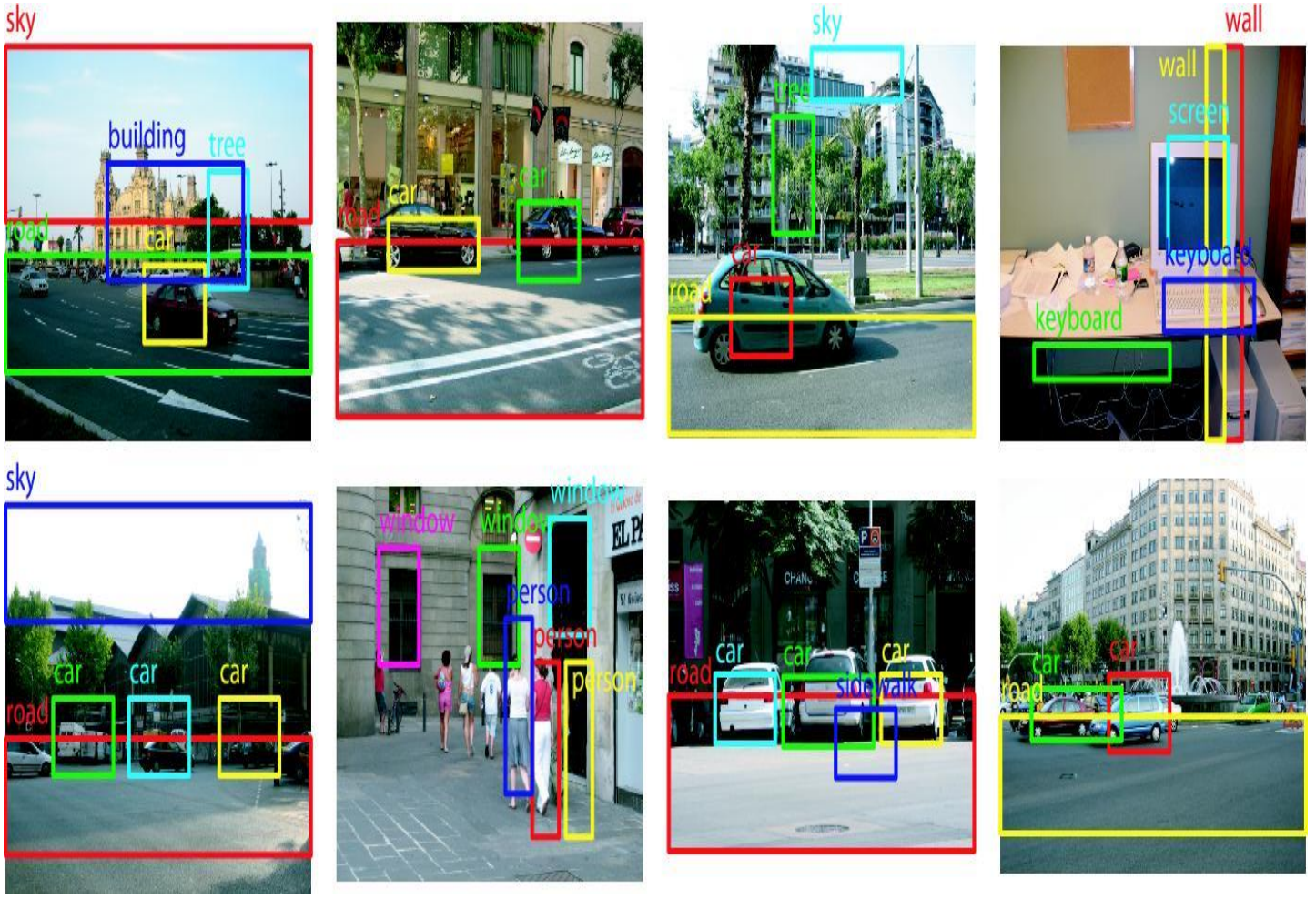
Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms



Object Detection

Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms

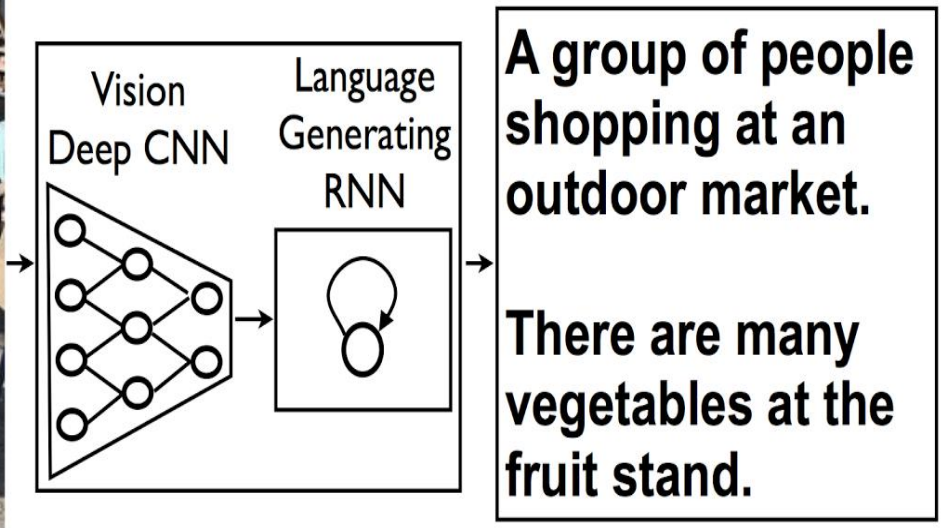


Image Captioning

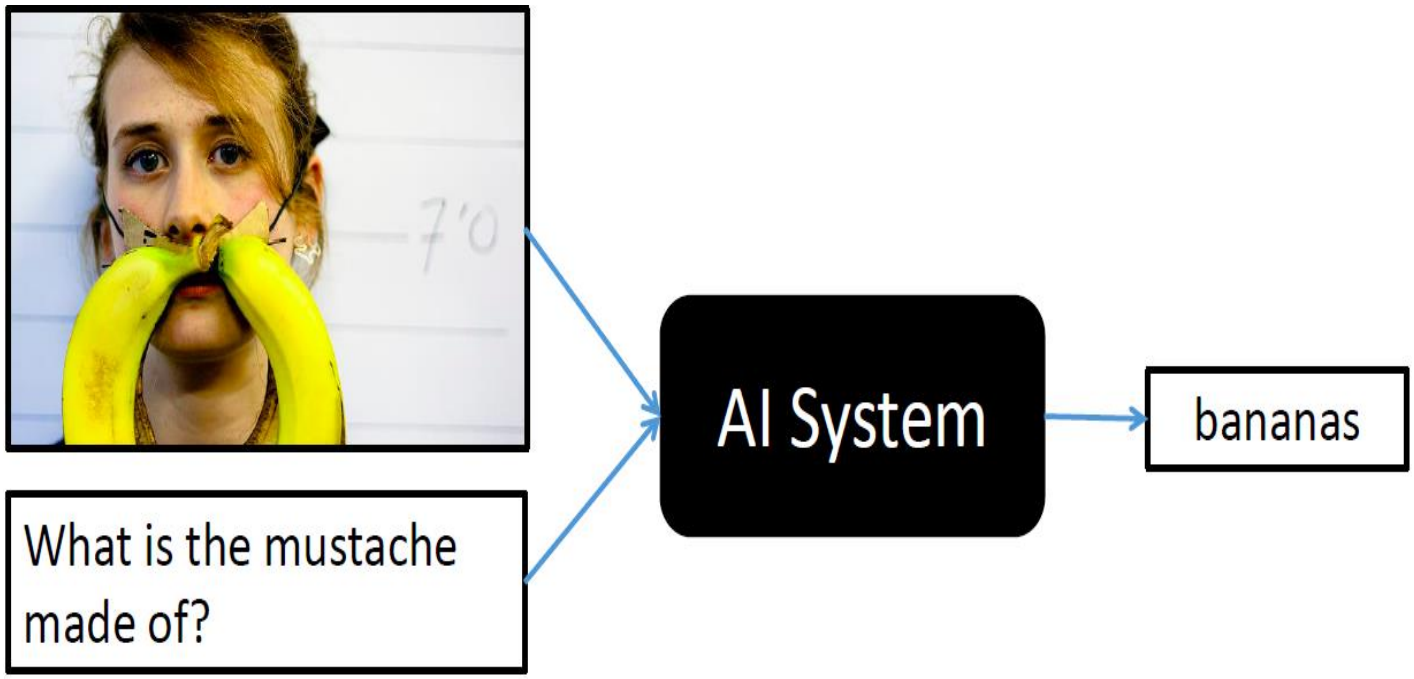
Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms



Visual Question Answering

Computer Vision

Tasks

↓

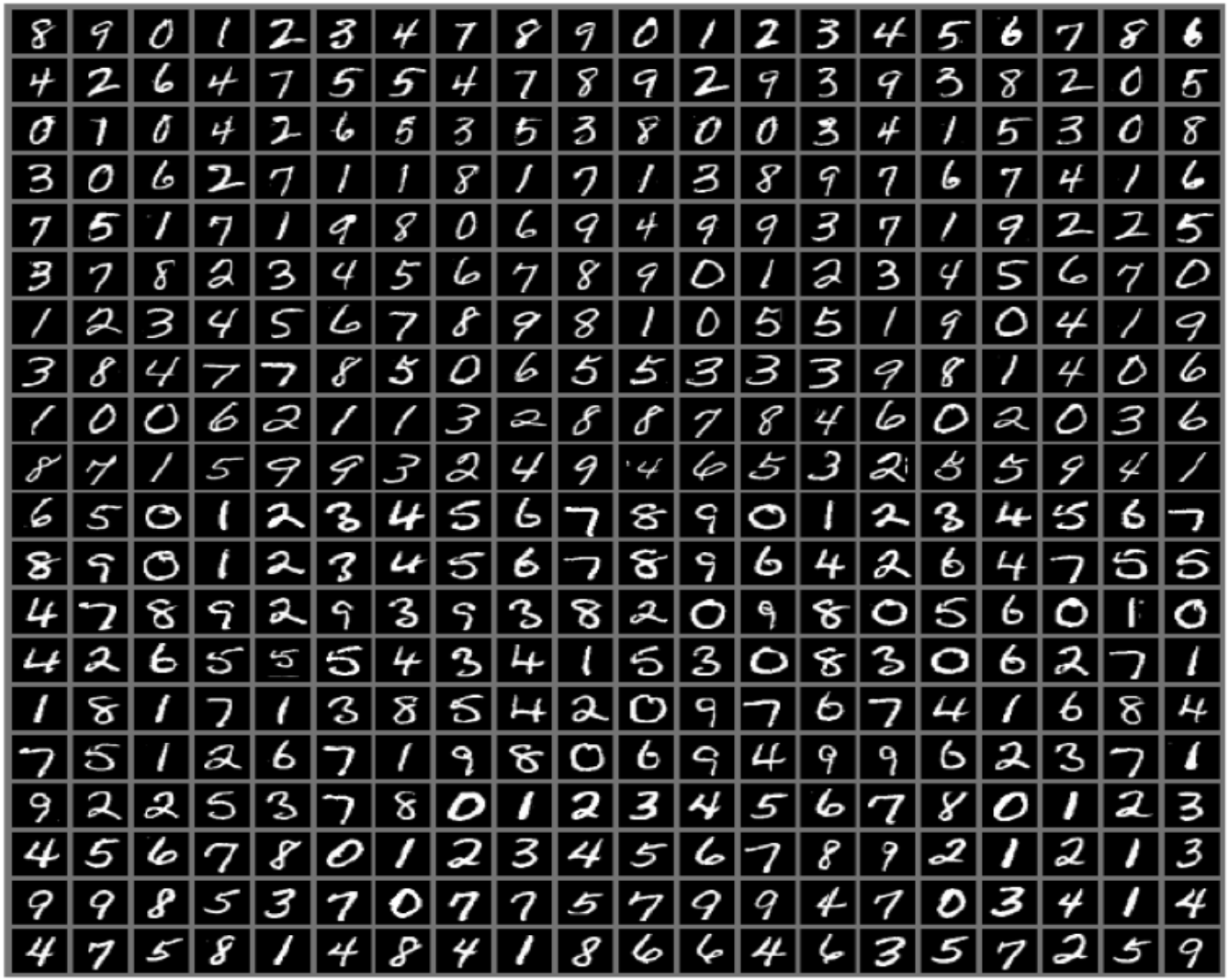
Datasets

↓

Feature Engineering

↓

Machine Learning Algorithms



MNIST (1998)

Computer Vision

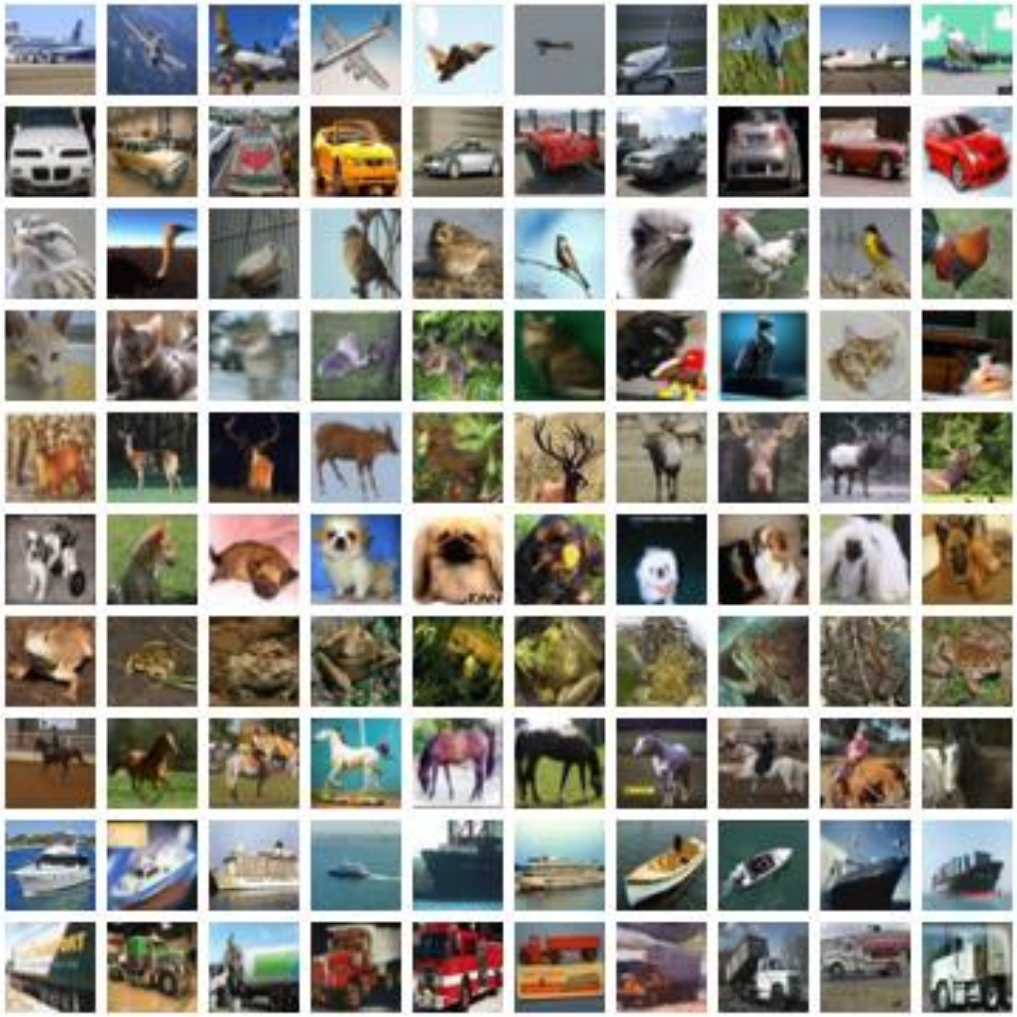
Tasks

Datasets

Feature Engineering

Machine Learning Algorithms

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck



CIFAR10 & CIFAR100

Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms

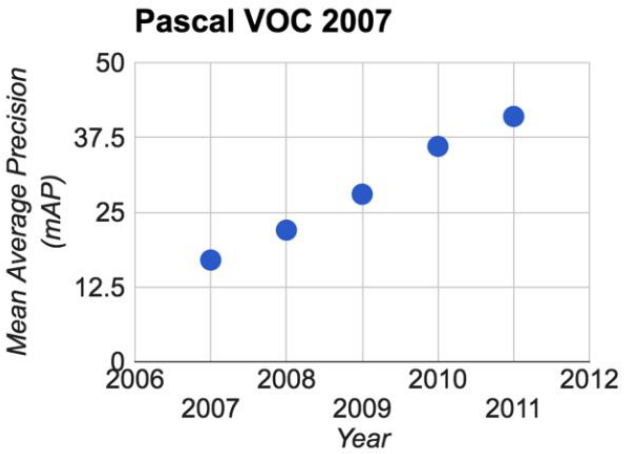
PASCAL Visual Object Challenge (20 object categories)

[Everingham et al. 2006-2012]

Image is CC BY-SA 3.0

Image is CC0 1.0 public domain

This image is licensed under CC BY-SA 2.0; changes made



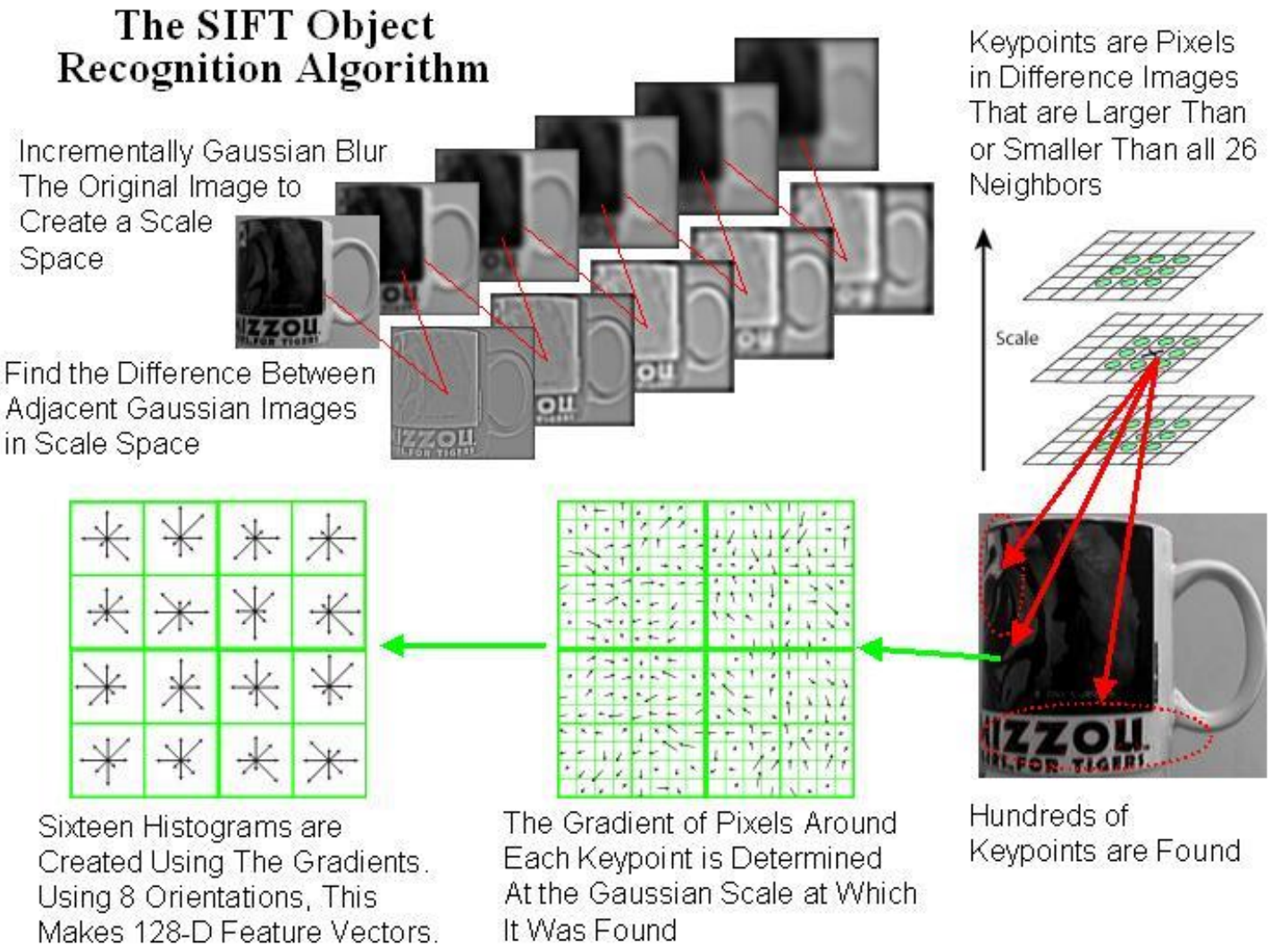
Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms



SIFT Feature (1999) http://weitz.de/sift/index.html?size=large

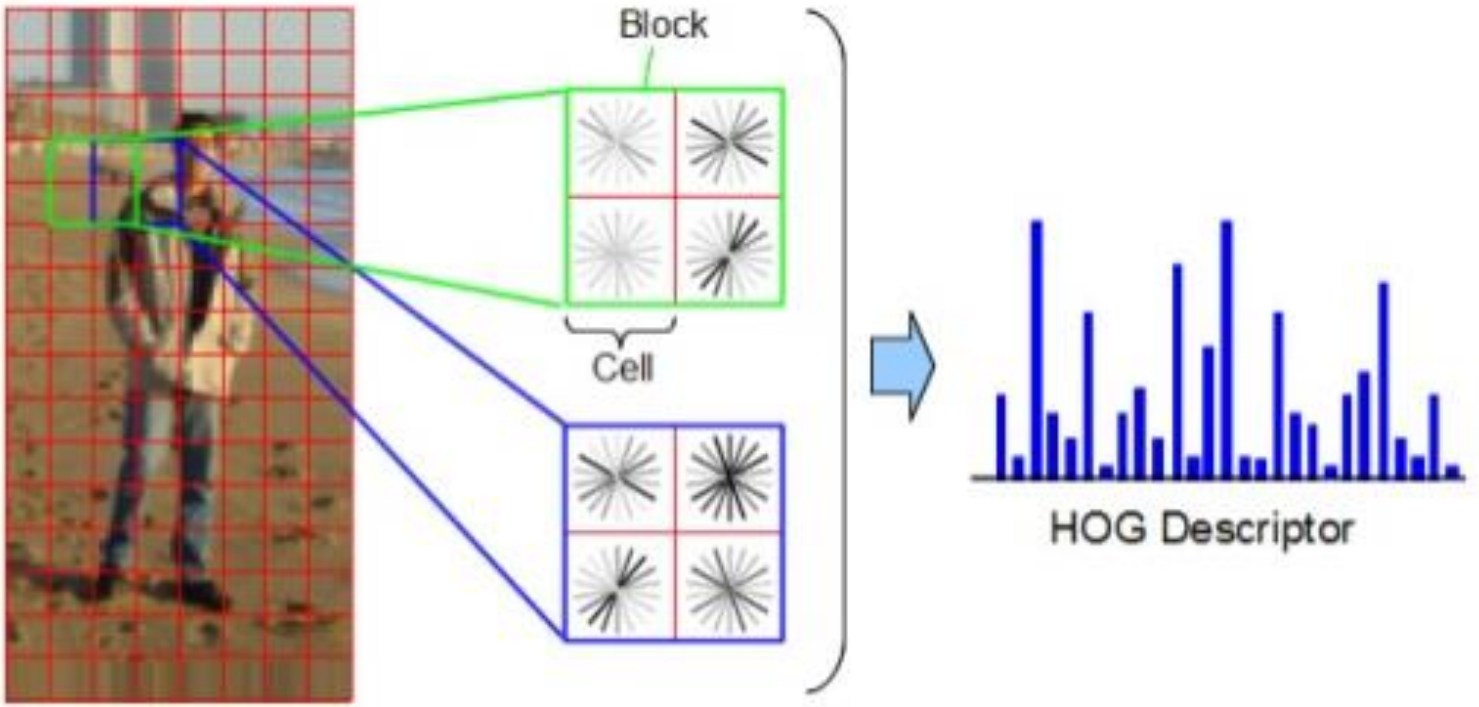
Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms



Histogram of oriented gradients (2005)

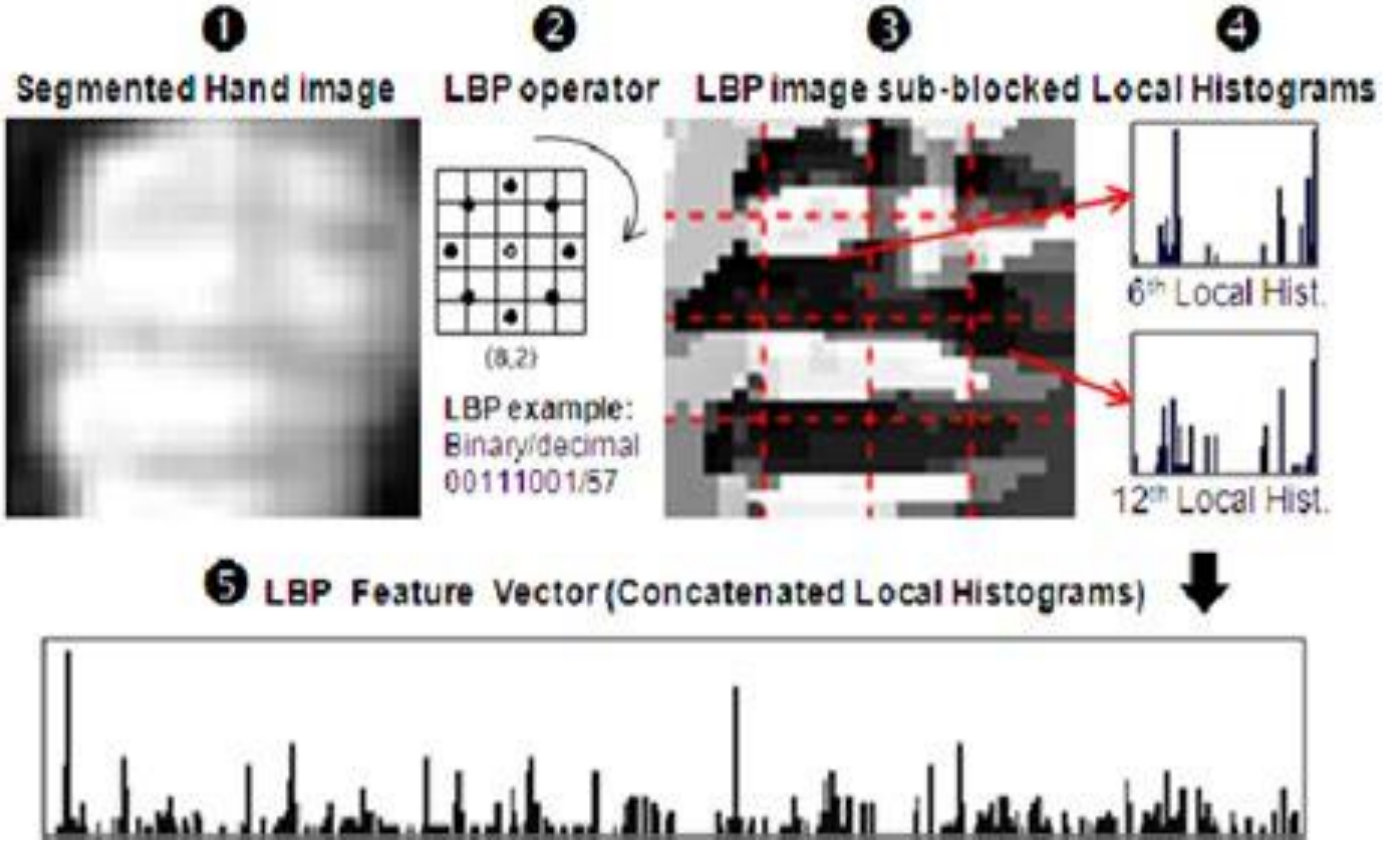
Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms



Local Binary Patterns (1994)

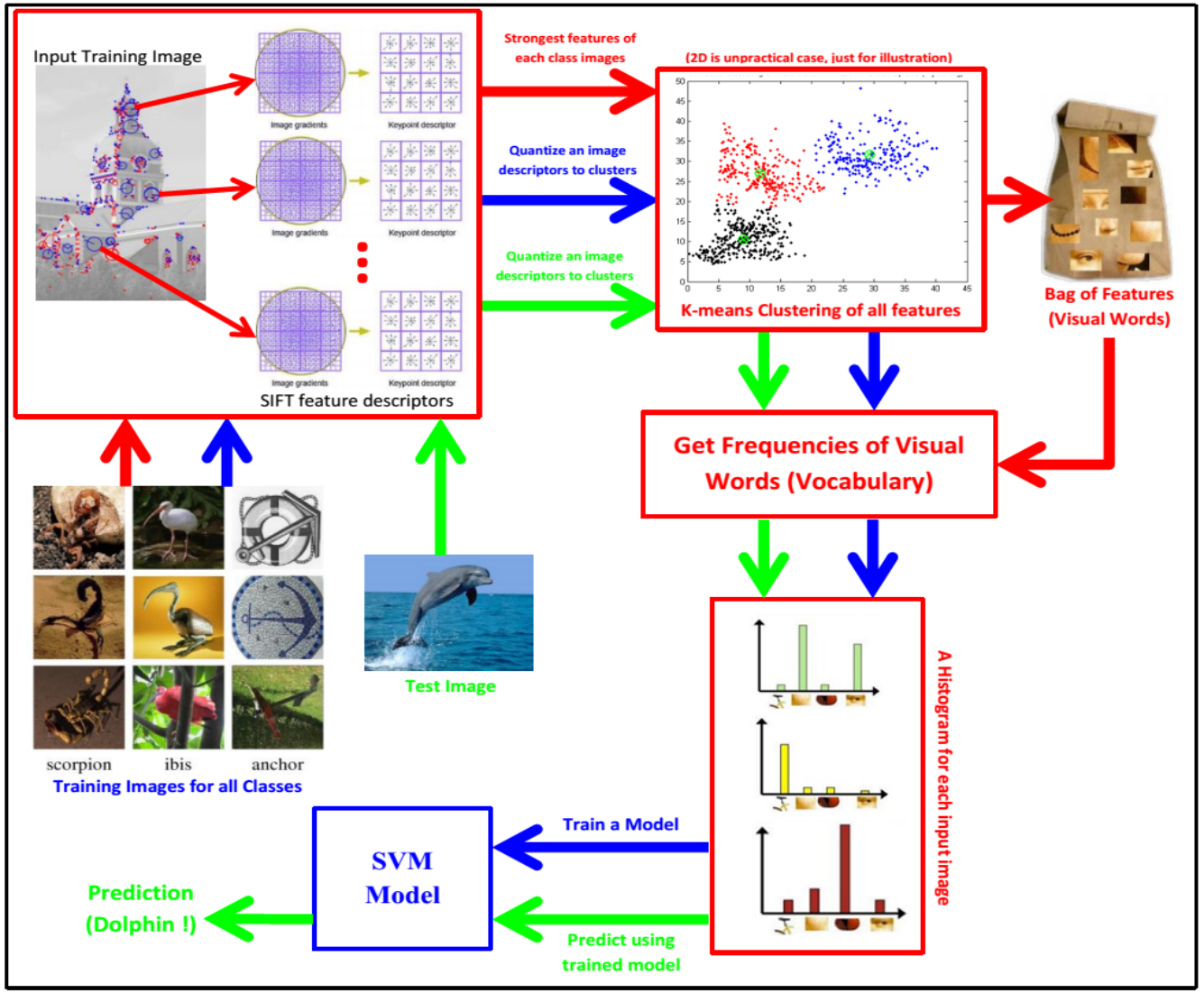
Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms



Bag-of-visual-words for image classification

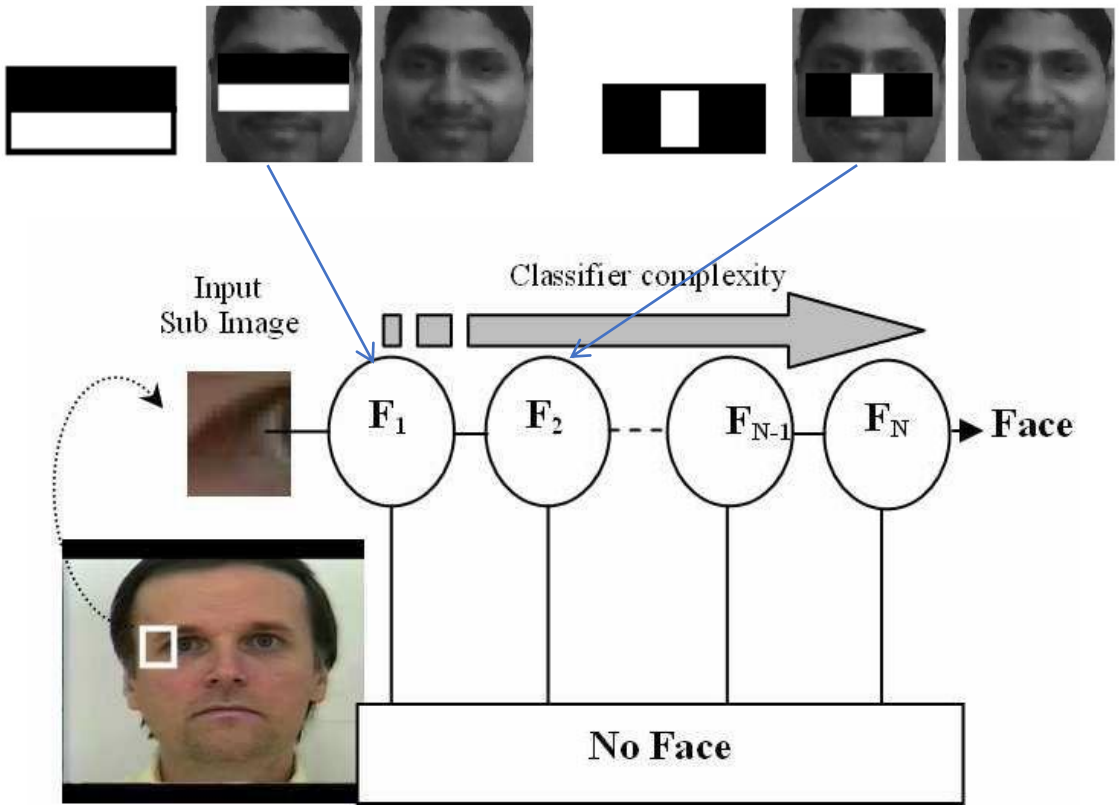
Computer Vision

Tasks

Datasets

Feature Engineering

Machine Learning Algorithms



Viola-Jones Real-time face detector (2001)

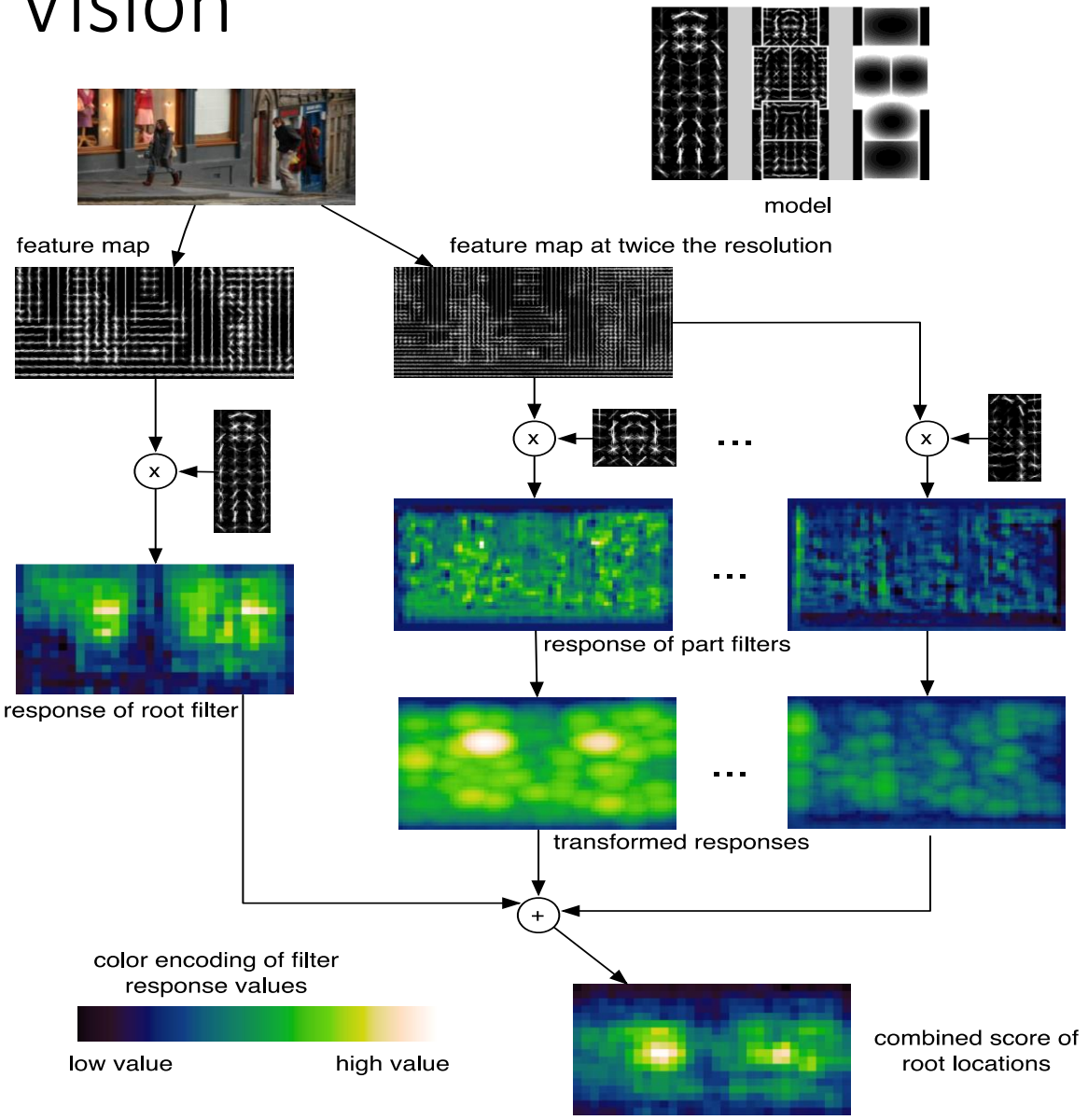
Computer Vision

Tasks

Datasets

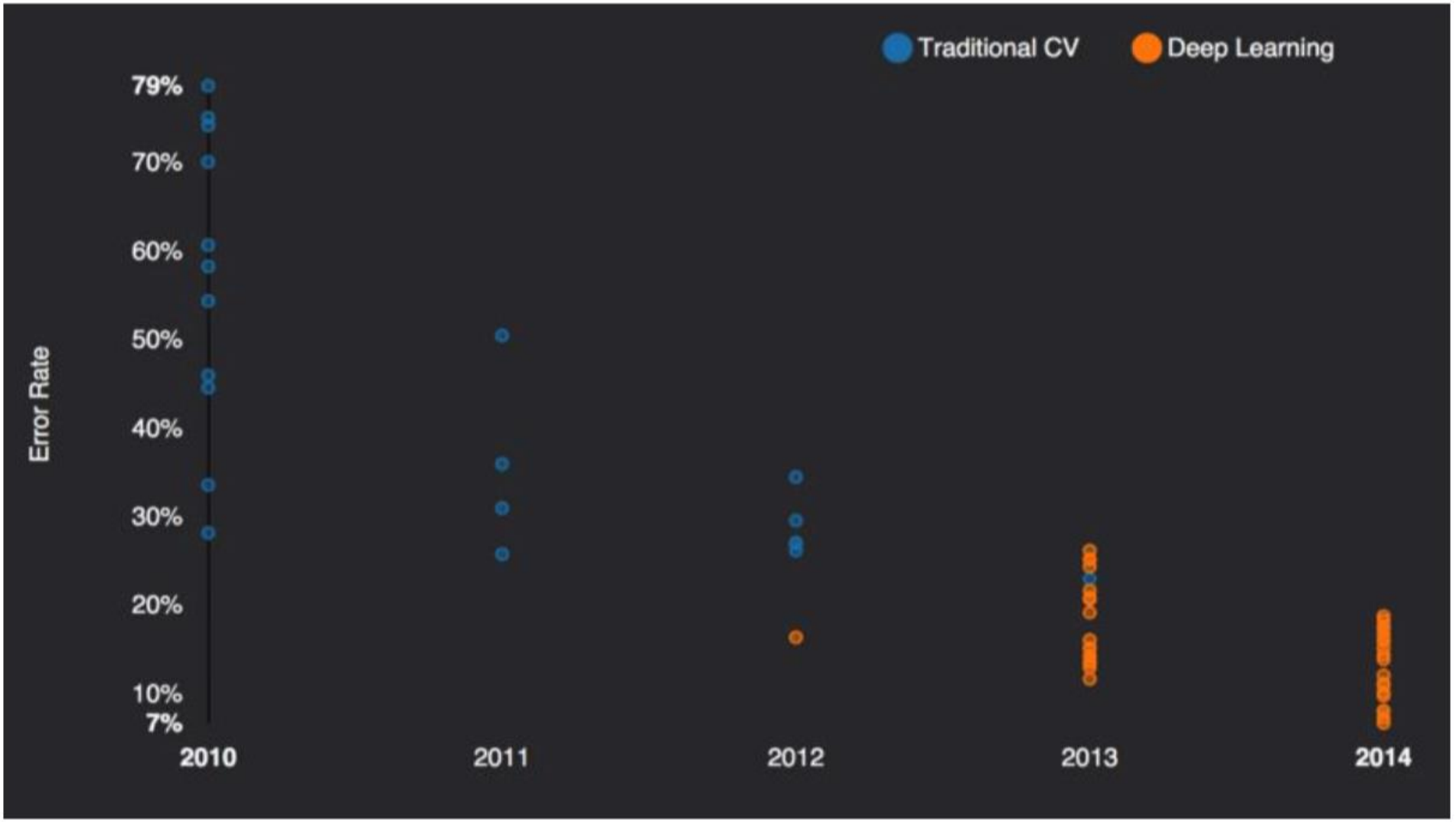
Feature Engineering

Machine Learning Algorithms



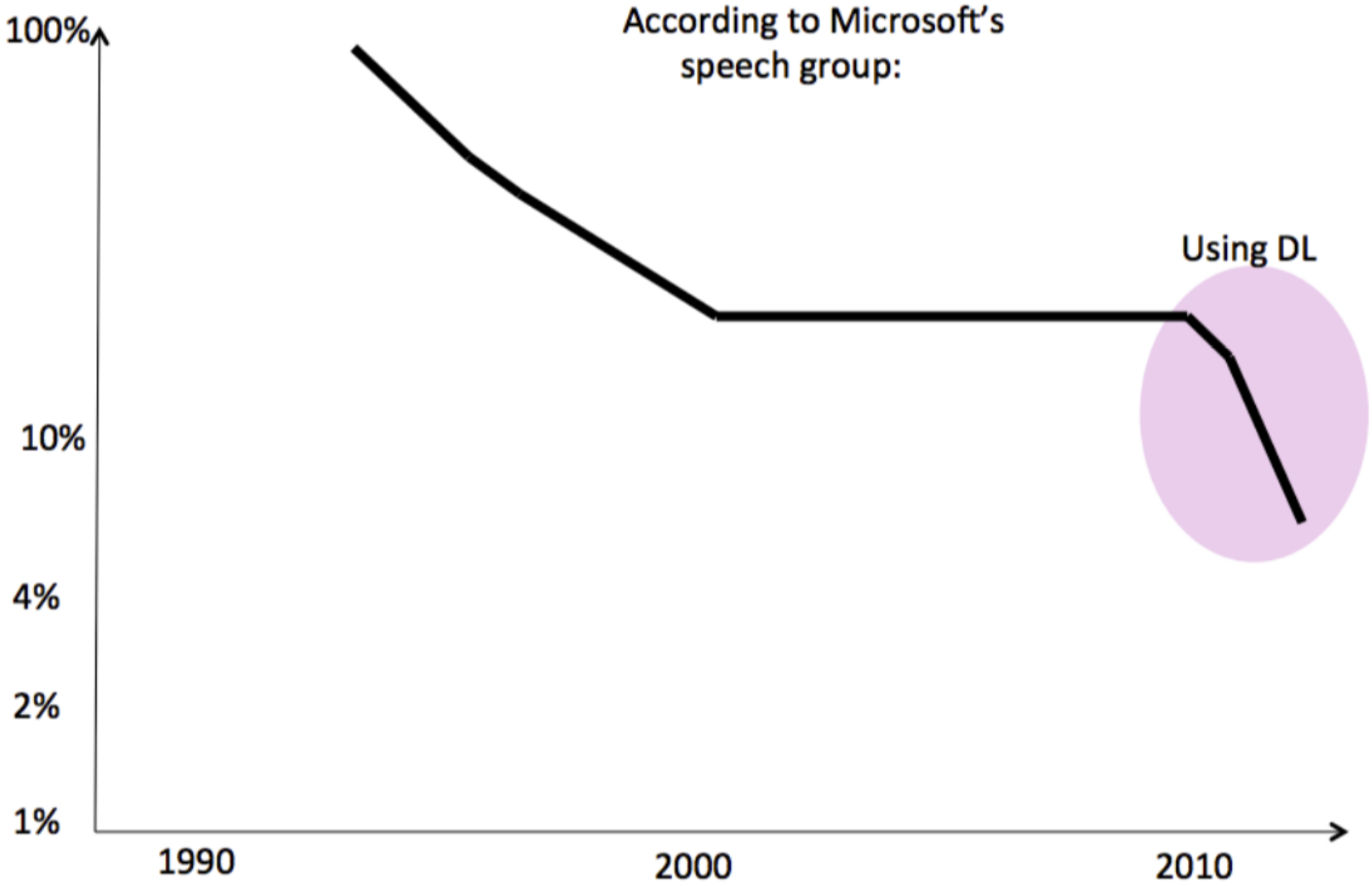
Deformable Part Model (2008)

Breakthroughs in 2012



ImageNet: The "computer vision World Cup"

Breakthroughs in 2012



Deep Learning in Speech Recognition

IMAGENET

www.image-net.org

22K categories and 14M images

- Animals
  - Bird
  - Fish
  - Mammal
  - Invertebrate
- Plants
  - Tree
  - Flower
- Food
- Materials
- Structures
- Artifact
  - Tools
  - Appliances
  - Structures
- Person
- Scenes
  - Indoor
  - Geological Formations
- Sport Activities

Deng, Dong, Socher, Li, Li, & Fei-Fei, 2009

IMAGENET Large Scale Visual Recognition Challenge

Steel drum

The Image Classification Challenge:

1,000 object classes

1,431,167 images

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

✔

Output:
Scale
T-shirt
Giant panda
Drumstick
Mud turtle

✗

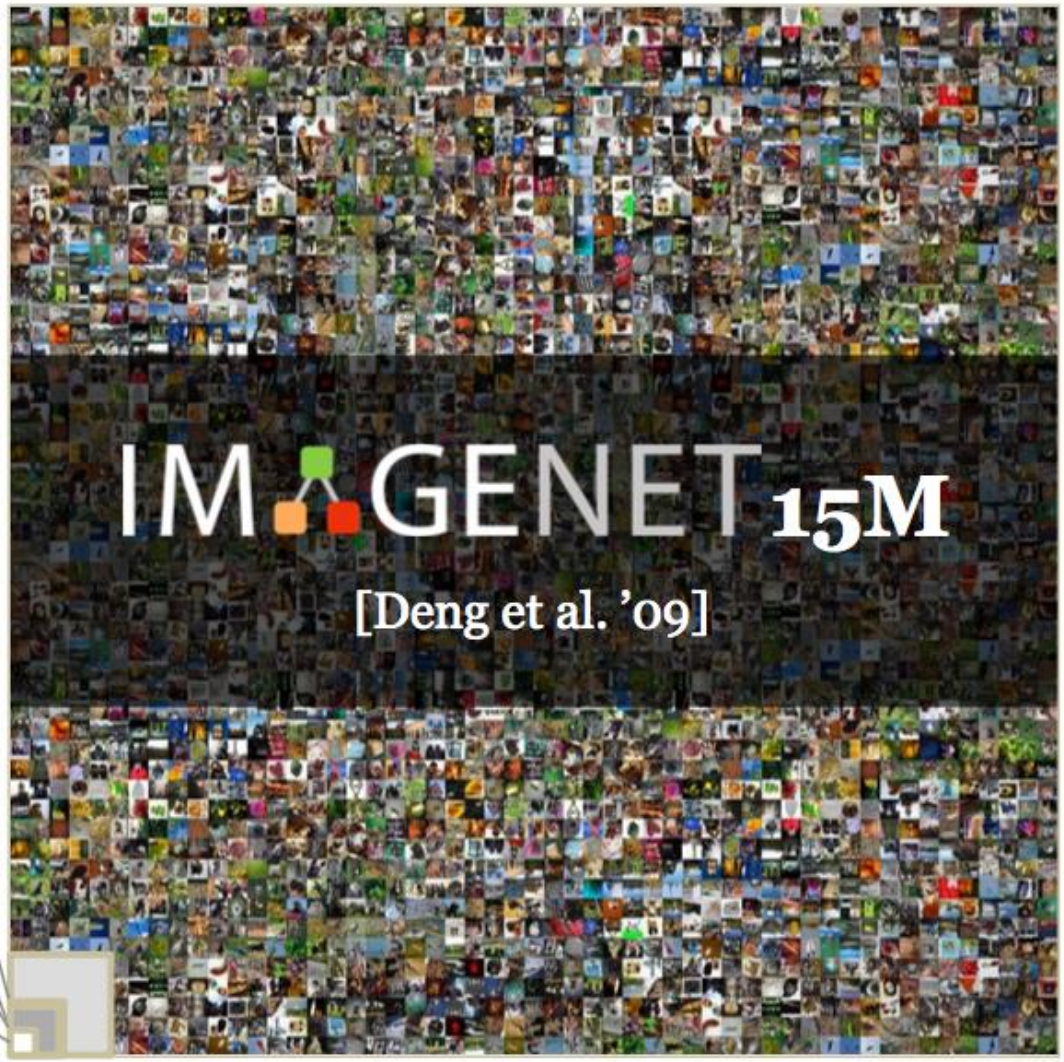
Russakovsky et al. arXiv, 2014

SUN, 131K
[Xiao et al. '10]

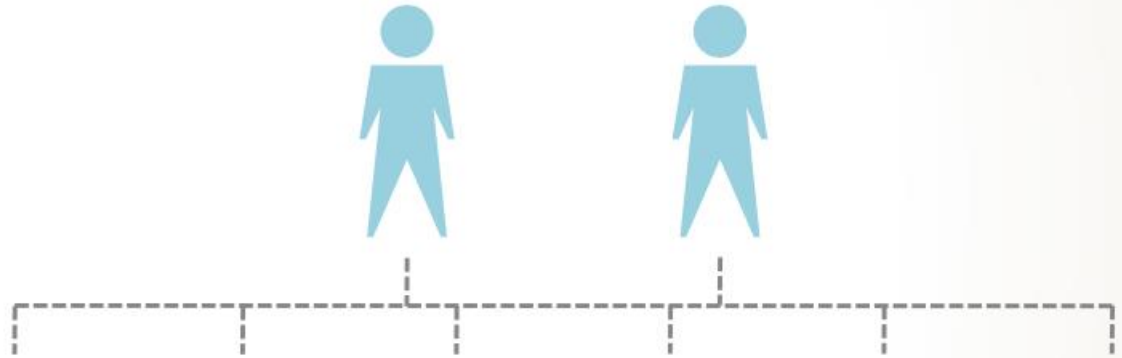
LabelMe, 37K
[Russell et al. '07]

PASCAL VOC, 30K
[Everingham et al. '06-'12]

Caltech101, 9K
[Fei-Fei, Fergus, Perona, '03]



ImageNet PhD Students



Crowdsourced Labor

amazon mechanical turk™
Artificial Artificial Intelligence

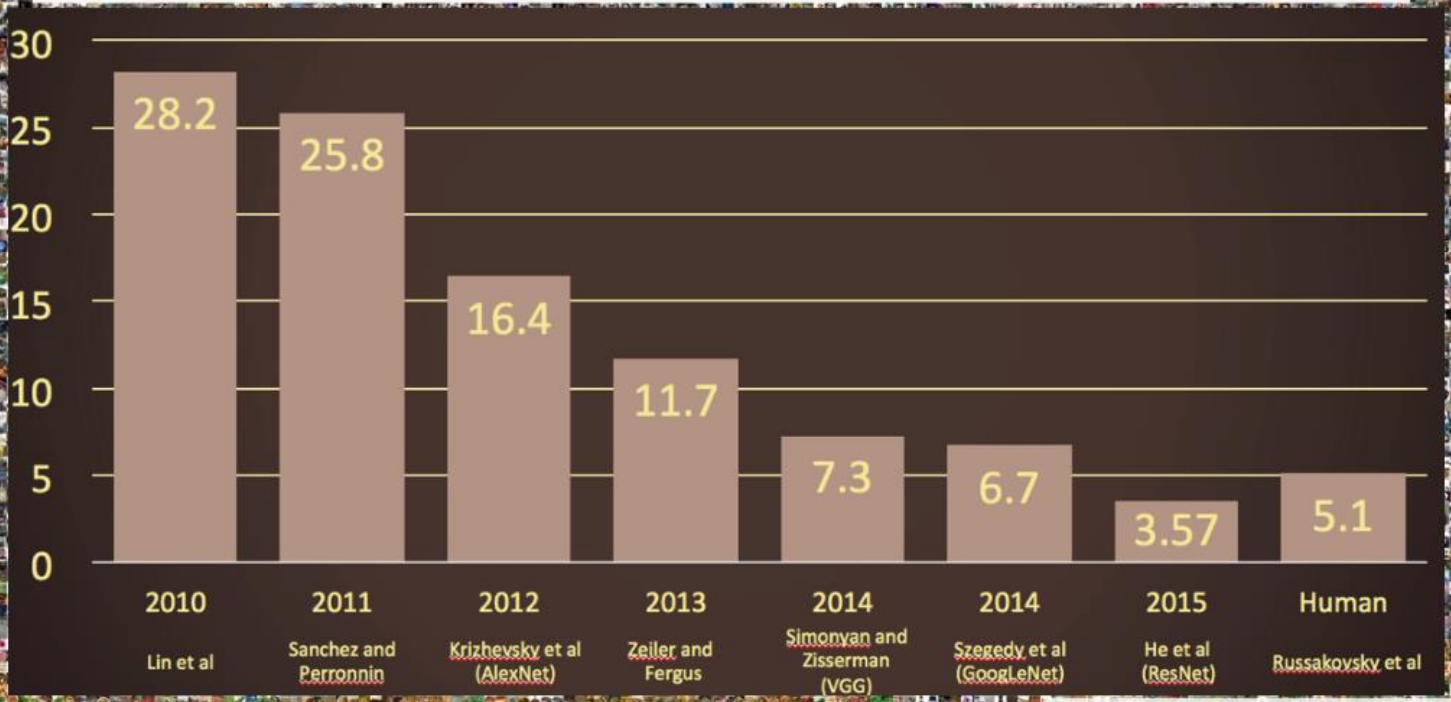
49k Workers *from* 167 Countries
2007-2010

IMAGENET Large Scale Visual Recognition Challenge

The Image Classification Challenge:

1,000 object classes

1,431,167 images



Russakovsky et al. arXiv, 2014

How Humans Compare

Human

5.1%
Top-5 error rate

Susceptible to:

- Fine-grained recognition
- Class unawareness
- Insufficient training data

GoogLeNet

6.8%
Top-5 error rate

Susceptible to:

- Small, thin objects
- Image filters
- Abstract representations
- Miscellaneous sources

IMAGENET Large Scale Visual Recognition Challenge

Year 2010

NEC-UIUC

Dense descriptor grid: HOG, LBP

Coding: local coordinate, super-vector

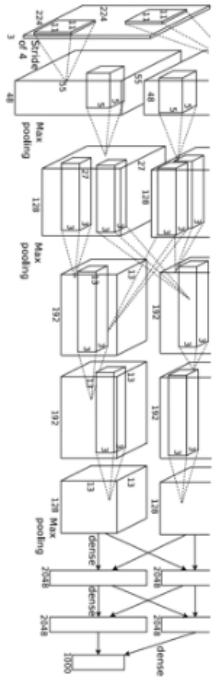
Pooling, SPM

Linear SVM

[Lin CVPR 2011]

Year 2012

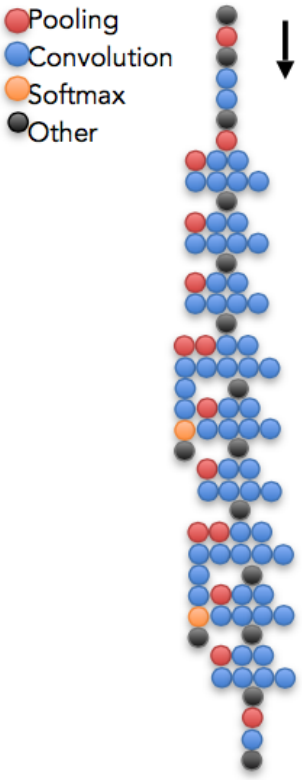
SuperVision



[Krizhevsky NIPS 2012]

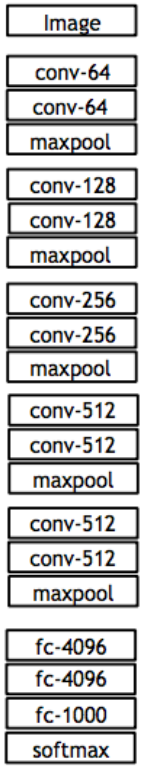
Year 2014

GoogLeNet



[Szegedy arxiv 2014]

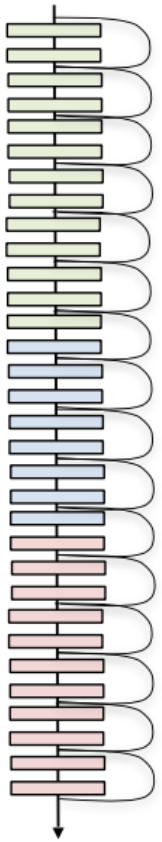
VGG



[Simonyan arxiv 2014]

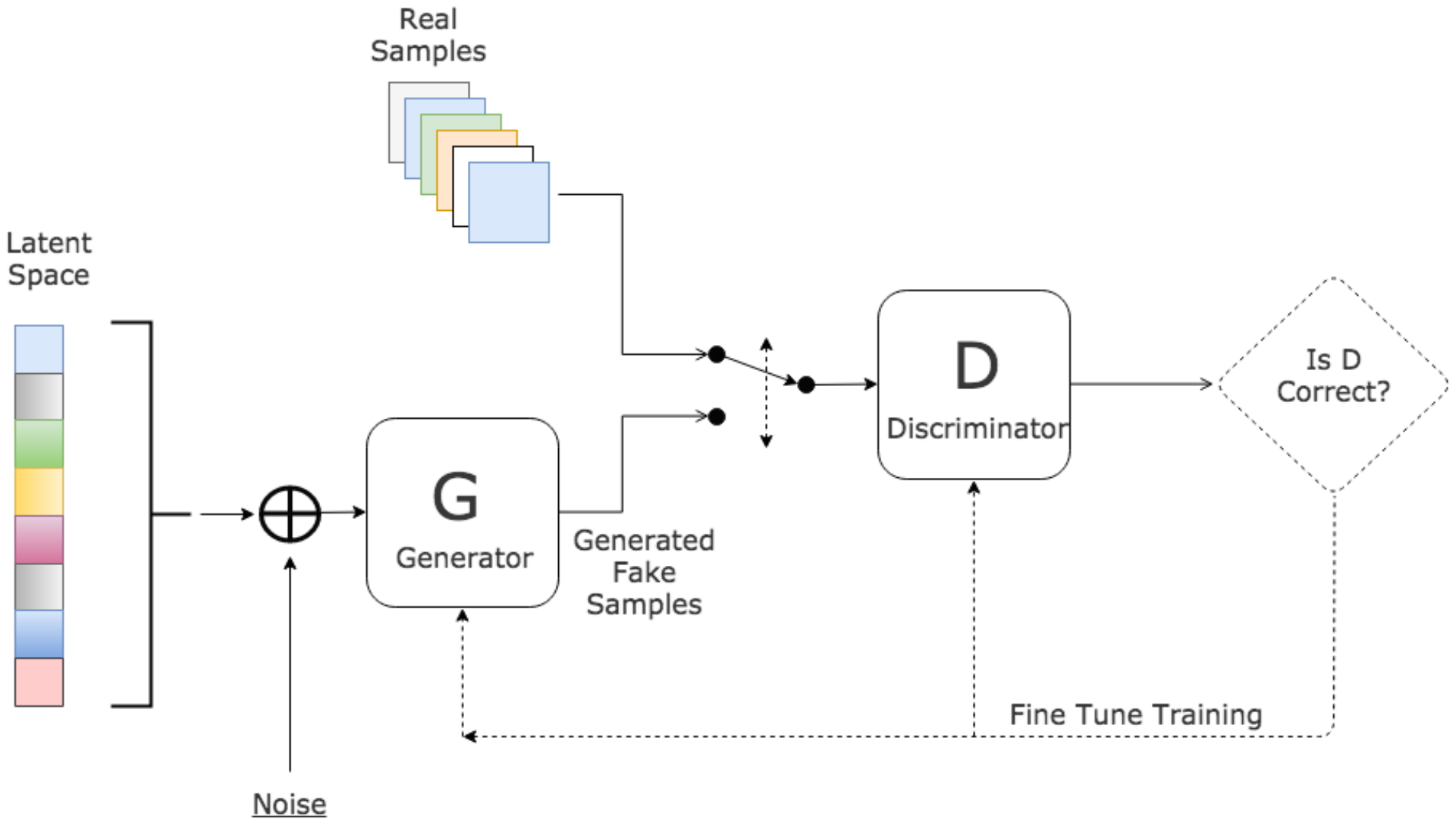
Year 2015

MSRA

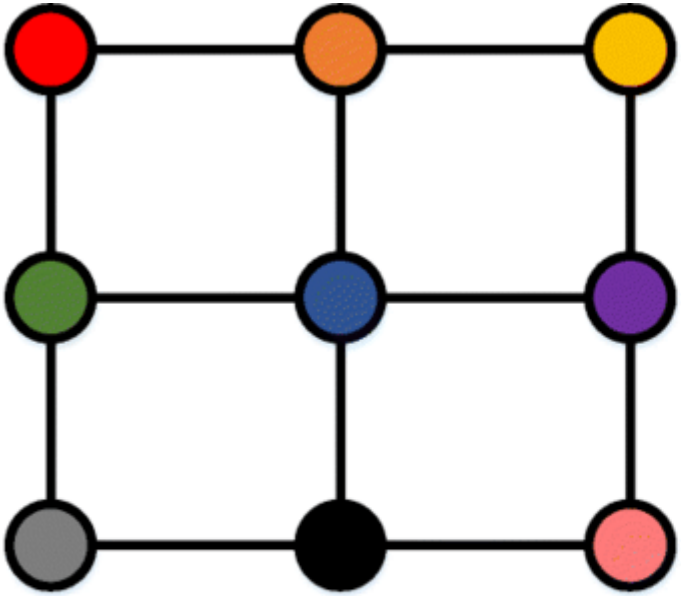


[He ICCV 2015]

Generative Adversarial Network (Ian Goodfellow, 2014)

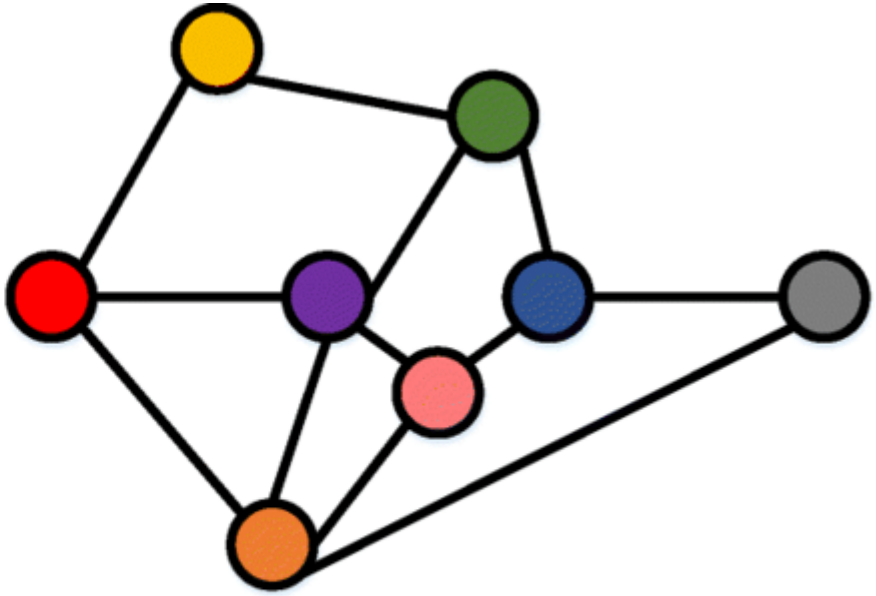


Graph Neural Networks



CNN

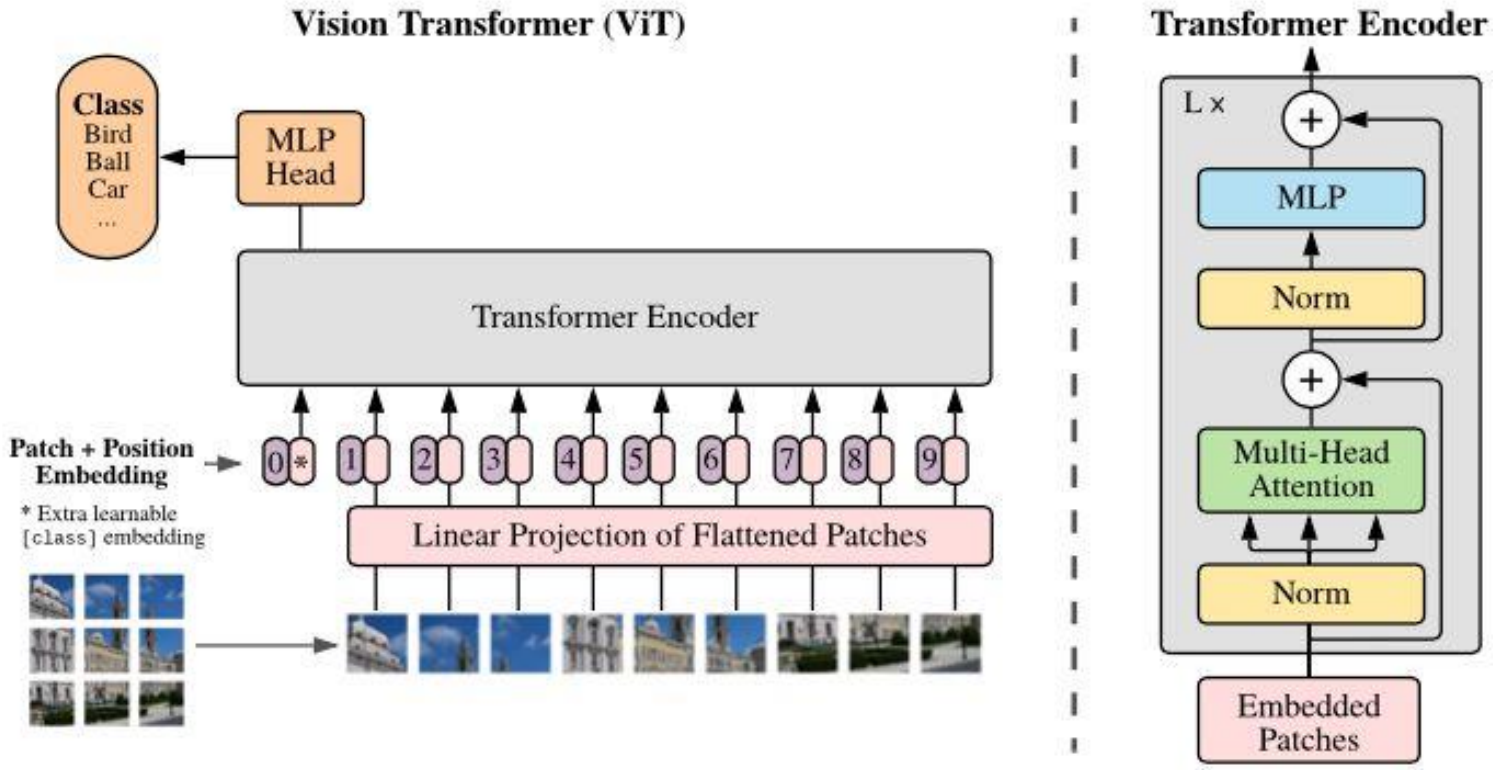
In Euclidean Space



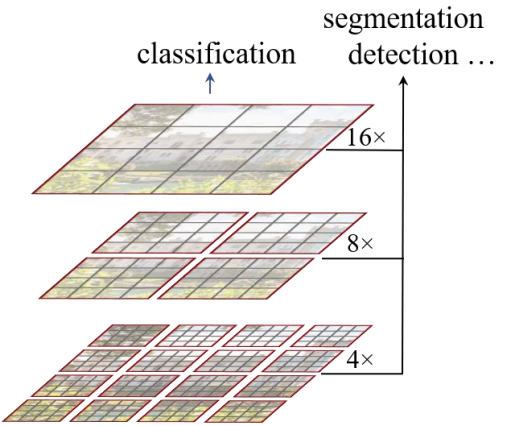
GNN

In Non-Euclidean Space

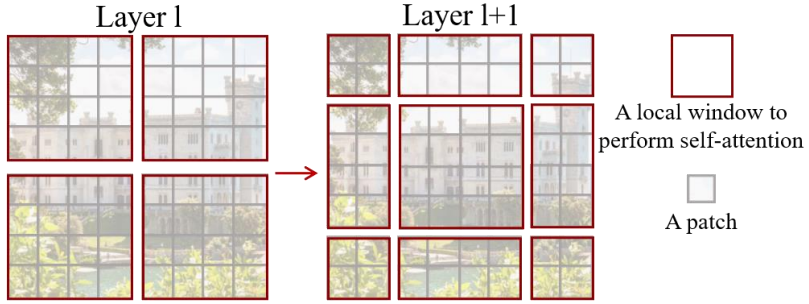
Vision Transformer (A. Dosovitskiy, 2020)



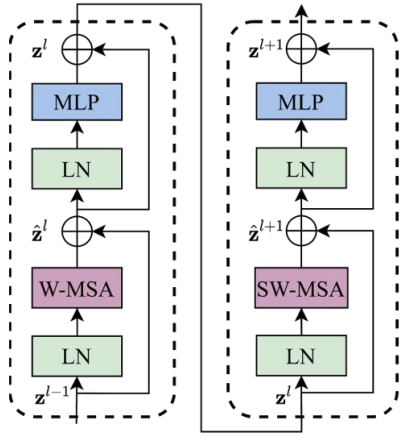
Hierarchical Vision Transformer (Z. Liu, 2021)



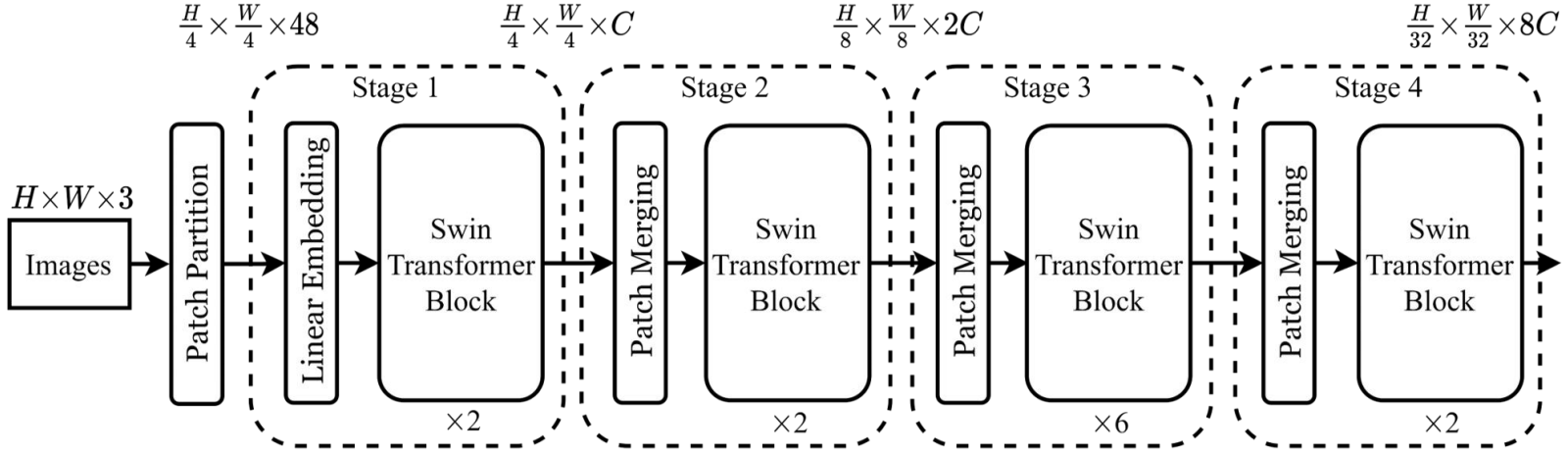
(a) Swin Transformer



(b) Shifted Window

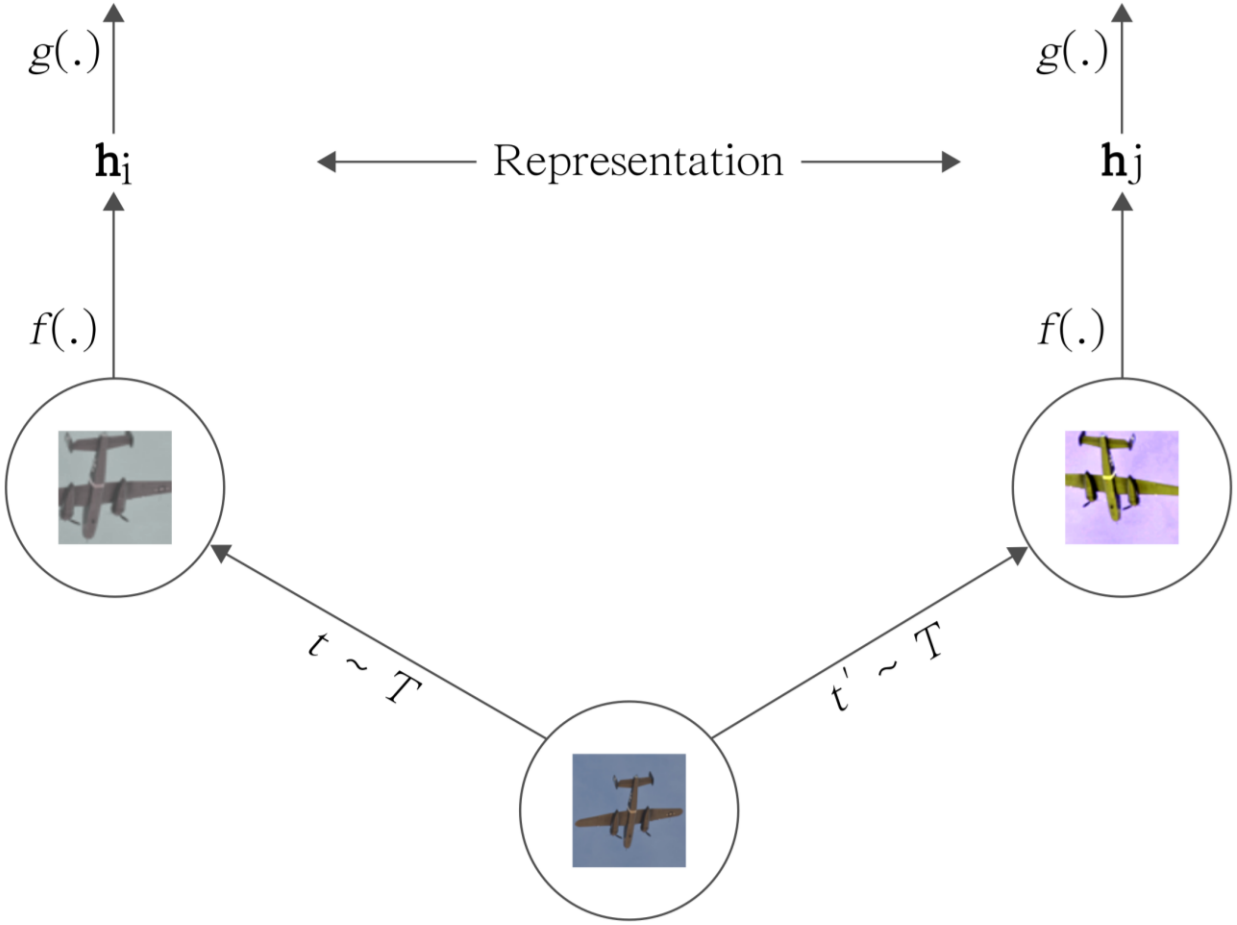


(c) Two Successive Swin Transformer Blocks

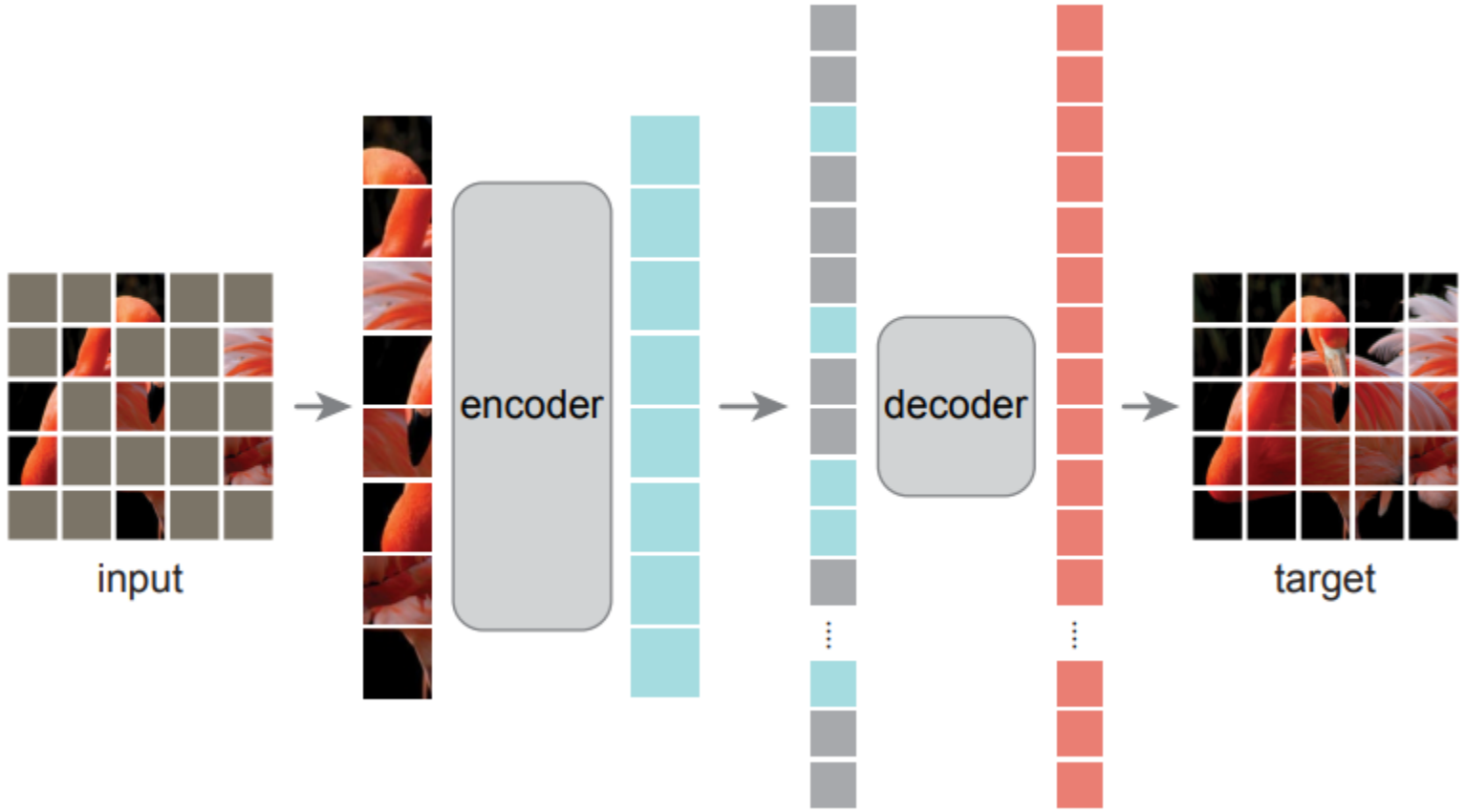


(d) Architecture

Self-supervised Learning (Simclr, T. Chen, 2021)

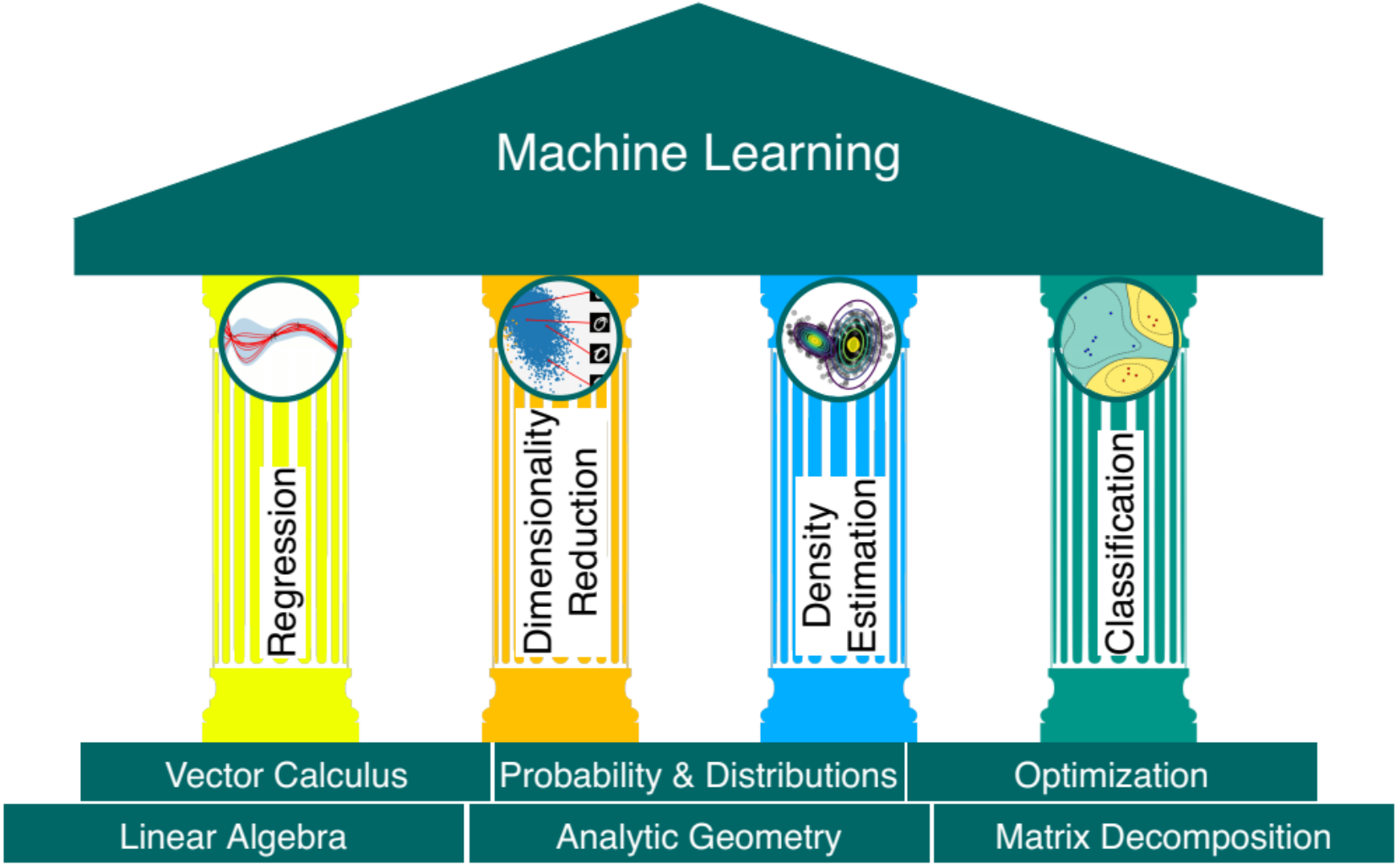


Self-supervised Learning (Masked Autoencoders, K. He, 2021)



Lecture One
Mathematics for AI

The foundations of Machine Learning



Main concept of Machine Learning

- We represent data as **vectors**.
- We choose an appropriate model, either using the **probabilistic or optimization** view.
- We learn from available data by using **numerical optimization** methods with the aim that the model performs well on data not used for training.

Essential Mathematical Concepts

Linear Algebra

We represent numerical data as vectors and represent a table of such data as a matrix. The study of vectors and matrices is called *linear algebra.*

- **Vectors**
  definition, scalars, addition, scalar multiplication, inner product(dot product), vector projection, cosine similarity, orthogonal vectors, normal and orthonormal vectors, vector norm, vector space, linear combination, linear span, linear independence, basis vectors
- **Matrices**
  definition, addition, transpose, scalar multiplication, matrix multiplication, matrix multiplication properties, hadamard product, functions, linear transformation, determinant, identity matrix, invertible matrix and inverse, rank, trace, popular type of matrices-symmetric, diagonal, orthogonal, orthonormal, positive definite matrix
- **Eigenvalues & eigenvectors**
  concept, intuition, significance, how to find
- **Principle component analysis**
  concept, properties, applications
- **Singular value decomposition**
  concept, properties, applications

Essential Mathematical Concepts

Calculus and Optimization

To train machine learning models, we typically find parameters that maximize some performance measure. Many optimization techniques require the concept of a gradient, which tells us the direction in which to search for a solution. *Vector calculus* is about gradient computation, and *optimization* is to find maxima/minima of functions.

- **Functions**
- **Scalar derivative**
  definition, intuition, common rules of differentiation, chain rule, partial derivatives
- **Gradient**
  concept, intuition, properties, directional derivative
- **Vector and matrix calculus**
  how to find derivative of {scalar-valued, vector-valued} function wrt a {scalar, vector} -> four combinations (Jacobian)
- **Gradient algorithms**
  local/global maxima and minima, saddle point, convex functions, gradient descent algorithms (batch, mini-batch, stochastic, their performance comparison)

Essential Mathematical Concepts

Probability

We often would also like to have predictors that allow us to express some sort of uncertainty, e.g., to quantify the confidence we have about the value of the prediction at a particular test data point. Quantification of uncertainty is the realm of *probability theory.*

- **Basic rules and axioms**

  events, sample space, frequentist approach, dependent and independent events, conditional probability

- **Random variables**

  continuous and discrete, expectation, variance, distributions- joint and conditional

- **Bayes' Theorem, MAP, MLE**
- **Popular distributions**

  binomial, bernoulli, poisson, exponential, gaussian

- **Conjugate priors**

Notation

$a, b, c, \alpha, \beta, \gamma$	Scalars are lowercase
$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$	Vectors are bold lowercase
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$	Matrices are bold uppercase
$\boldsymbol{x}^\top, \boldsymbol{A}^\top$	Transpose of a vector or matrix
$\boldsymbol{A}^{-1}$	Inverse of a matrix
$\langle \boldsymbol{x}, \boldsymbol{y} \rangle$	Inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$
$\boldsymbol{x}^\top \boldsymbol{y}$	Dot product of $\boldsymbol{x}$ and $\boldsymbol{y}$
$B = (\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3)$	(Ordered) tuple
$\boldsymbol{B} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3]$	Matrix of column vectors stacked horizontally
$\mathcal{B} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$	Set of vectors (unordered)
$\mathbb{Z}, \mathbb{N}$	Integers and natural numbers, respectively
$\mathbb{R}, \mathbb{C}$	Real and complex numbers, respectively
$\mathbb{R}^n$	n-dimensional vector space of real numbers

Notation

$\forall x$	Universal quantifier: for all x
$\exists x$	Existential quantifier: there exists x
$a := b$	a is defined as b
$a =: b$	b is defined as a
$a \propto b$	a is proportional to b, i.e., $a = \text{constant} \cdot b$
$g \circ f$	Function composition: "g after f"
$\iff$	If and only if
$\implies$	Implies
$\mathcal{A}, \mathcal{C}$	Sets
$a \in \mathcal{A}$	a is an element of set $\mathcal{A}$
$\emptyset$	Empty set
$\mathcal{A} \backslash \mathcal{B}$	$\mathcal{A}$ without $\mathcal{B}$: the set of elements in $\mathcal{A}$ but not in $\mathcal{B}$
D	Number of dimensions; indexed by $d = 1, \ldots, D$
N	Number of data points; indexed by $n = 1, \ldots, N$

Notation

$\boldsymbol{I}_m$	Identity matrix of size $m \times m$
$\mathbf{0}_{m,n}$	Matrix of zeros of size $m \times n$
$\mathbf{1}_{m,n}$	Matrix of ones of size $m \times n$
$\boldsymbol{e}_i$	Standard/canonical vector (where i is the component that is 1)
$\dim$	Dimensionality of vector space
$\mathrm{rk}(\boldsymbol{A})$	Rank of matrix $\boldsymbol{A}$
$\mathrm{Im}(\Phi)$	Image of linear mapping Φ
$\ker(\Phi)$	Kernel (null space) of a linear mapping Φ
$\mathrm{span}[\boldsymbol{b}_1]$	Span (generating set) of $\boldsymbol{b}_1$
$\mathrm{tr}(\boldsymbol{A})$	Trace of $\boldsymbol{A}$
$\det(\boldsymbol{A})$	Determinant of $\boldsymbol{A}$
$\lvert \cdot \rvert$	Absolute value or determinant (depending on context)
$\lVert \cdot \rVert$	Norm; Euclidean, unless specified
λ	Eigenvalue or Lagrange multiplier
E_λ	Eigenspace corresponding to eigenvalue λ

Notation

$\boldsymbol{x} \perp \boldsymbol{y}$	Vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are orthogonal
V	Vector space
$V^\perp$	Orthogonal complement of vector space V
$\sum_{n=1}^{N} x_n$	Sum of the x_n: $x_1 + \ldots + x_N$
$\prod_{n=1}^{N} x_n$	Product of the x_n: $x_1 \cdot \ldots \cdot x_N$
$\boldsymbol{\theta}$	Parameter vector
$\frac{\partial f}{\partial x}$	Partial derivative of f with respect to x
$\frac{\mathrm{d}f}{\mathrm{d}x}$	Total derivative of f with respect to x
∇	Gradient
$f_* = \min_x f(x)$	The smallest function value of f
$x_* \in \arg\min_x f(x)$	The value x_* that minimizes f (note: $\arg\min$ returns a set of values)

Notation

$\mathfrak{L}$	Lagrangian
$\mathcal{L}$	Negative log-likelihood
$\binom{n}{k}$	Binomial coefficient, n choose k
$\mathbb{V}_X[\boldsymbol{x}]$	Variance of $\boldsymbol{x}$ with respect to the random variable X
$\mathbb{E}_X[\boldsymbol{x}]$	Expectation of $\boldsymbol{x}$ with respect to the random variable X
$\mathrm{Cov}_{X,Y}[\boldsymbol{x}, \boldsymbol{y}]$	Covariance between $\boldsymbol{x}$ and $\boldsymbol{y}$.
$X \perp\!\!\!\perp Y \mid Z$	X is conditionally independent of Y given Z
$X \sim p$	Random variable X is distributed according to p
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathrm{Ber}(\mu)$	Bernoulli distribution with parameter μ
$\mathrm{Bin}(N, \mu)$	Binomial distribution with parameters N, μ
$\mathrm{Beta}(\alpha, \beta)$	Beta distribution with parameters α, β

Notation

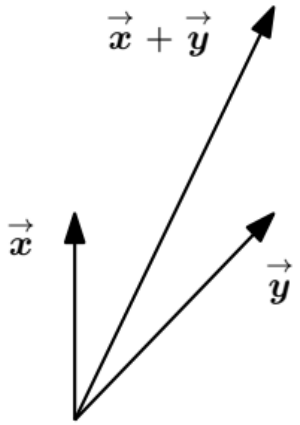
$\mathfrak{L}$	Lagrangian
$\mathcal{L}$	Negative log-likelihood
$\binom{n}{k}$	Binomial coefficient, n choose k
$\mathbb{V}_X[\boldsymbol{x}]$	Variance of $\boldsymbol{x}$ with respect to the random variable X
$\mathbb{E}_X[\boldsymbol{x}]$	Expectation of $\boldsymbol{x}$ with respect to the random variable X
$\mathrm{Cov}_{X,Y}[\boldsymbol{x}, \boldsymbol{y}]$	Covariance between $\boldsymbol{x}$ and $\boldsymbol{y}$.
$X \perp\!\!\!\perp Y \mid Z$	X is conditionally independent of Y given Z
$X \sim p$	Random variable X is distributed according to p
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathrm{Ber}(\mu)$	Bernoulli distribution with parameter μ
$\mathrm{Bin}(N, \mu)$	Binomial distribution with parameters N, μ
$\mathrm{Beta}(\alpha, \beta)$	Beta distribution with parameters α, β

Linear Algebra

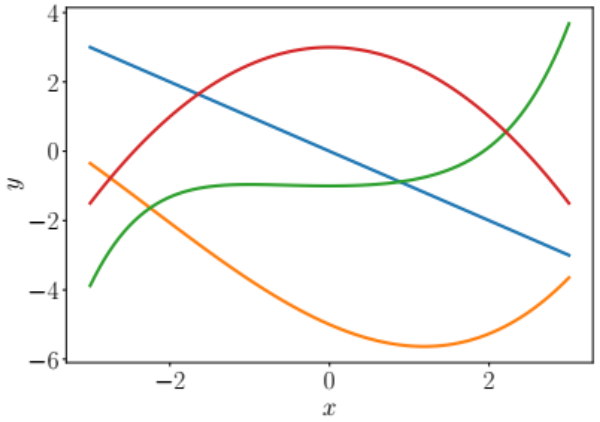
Vectors

In general, vectors are special objects that can be added together and multiplied by scalars to produce another object of the same kind. From an abstract mathematical viewpoint, any object that satisfies these two properties can be considered a vector.

1. Geometric vectors
2. Polynomials
3. Audio signals
4. Elements of R^n



(a) Geometric vectors.



(b) Polynomials.

Matrices: definition

Definition 2.1 (Matrix). With $m, n \in \mathbb{N}$ a real-valued (m, n) *matrix* $\boldsymbol{A}$ is an $m \cdot n$-tuple of elements a_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$, which is ordered according to a rectangular scheme consisting of m rows and n columns:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad a_{ij} \in \mathbb{R}. \tag{2.11}$$

By convention $(1, n)$-matrices are called *rows* and $(m, 1)$-matrices are called *columns*. These special matrices are also called *row/column vectors*.

Matrices: addition and multiplication

The sum of two matrices $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $\boldsymbol{B} \in \mathbb{R}^{m \times n}$ is defined as the element-wise sum, i.e.,

$$\boldsymbol{A} + \boldsymbol{B} := \begin{bmatrix} a_{11} + b_{11} & \cdots & a_{1n} + b_{1n} \\ \vdots & & \vdots \\ a_{m1} + b_{m1} & \cdots & a_{mn} + b_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n} . \tag{2.12}$$

For matrices $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $\boldsymbol{B} \in \mathbb{R}^{n \times k}$, the elements c_{ij} of the product $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B} \in \mathbb{R}^{m \times k}$ are computed as

$$c_{ij} = \sum_{l=1}^{n} a_{il} b_{lj}, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, k. \tag{2.13}$$

Q: What is the computational complexity of matrix addition and multiplication?

Matrices: addition and multiplication

The sum of two matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{B} \in \mathbb{R}^{m\times n}$ is defined as the element-wise sum, i.e.,

$$\boldsymbol{A} + \boldsymbol{B} := \begin{bmatrix} a_{11} + b_{11} & \cdots & a_{1n} + b_{1n} \\ \vdots & & \vdots \\ a_{m1} + b_{m1} & \cdots & a_{mn} + b_{mn} \end{bmatrix} \in \mathbb{R}^{m\times n}. \tag{2.12}$$

For matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{B} \in \mathbb{R}^{n\times k}$, the elements c_{ij} of the product $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B} \in \mathbb{R}^{m\times k}$ are computed as

$$c_{ij} = \sum_{l=1}^{n} a_{il} b_{lj}, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, k. \tag{2.13}$$

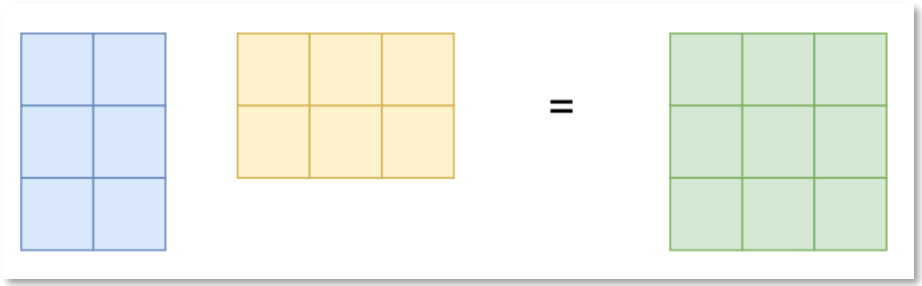
Q: What is the computational complexity of matrix addition and multiplication? O(mn) and O(mnk)

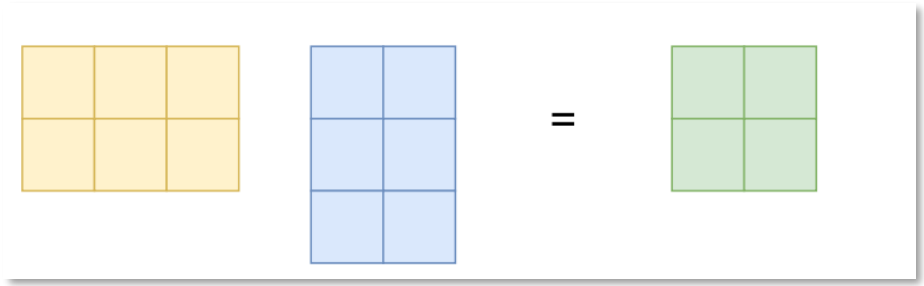
Matrices: addition and multiplication

Remark1: Matrices can only be multiplied if their "neighboring" dimensions match.

Remark2: Matrix multiplication is not defined as an element-wise operation on matrix elements. This kind of element-wise multiplication is called a Hadamard product.

Remark3: Even if both matrix multiplications **AB** and **BA** are defined, the dimensions of the results can be different.





Matrices: addition and multiplication

- *Associativity:*

$$\forall \boldsymbol{A} \in \mathbb{R}^{m \times n}, \boldsymbol{B} \in \mathbb{R}^{n \times p}, \boldsymbol{C} \in \mathbb{R}^{p \times q} : (\boldsymbol{AB})\boldsymbol{C} = \boldsymbol{A}(\boldsymbol{BC}) \tag{2.18}$$

- *Distributivity:*

$$\forall \boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}, \boldsymbol{C}, \boldsymbol{D} \in \mathbb{R}^{n \times p} : (\boldsymbol{A} + \boldsymbol{B})\boldsymbol{C} = \boldsymbol{AC} + \boldsymbol{BC} \tag{2.19a}$$

$$\boldsymbol{A}(\boldsymbol{C} + \boldsymbol{D}) = \boldsymbol{AC} + \boldsymbol{AD} \tag{2.19b}$$

- Multiplication with the identity matrix:

$$\forall \boldsymbol{A} \in \mathbb{R}^{m \times n} : \boldsymbol{I}_m \boldsymbol{A} = \boldsymbol{A} \boldsymbol{I}_n = \boldsymbol{A} \tag{2.20}$$

  Note that $\boldsymbol{I}_m \neq \boldsymbol{I}_n$ for $m \neq n$.

Matrices: identity matrix

Definition 2.2 (Identity Matrix). In $\mathbb{R}^{n\times n}$, we define the *identity matrix*

$$\boldsymbol{I}_n := \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{2.17}$$

Matrices: inverse and transpose

Definition 2.3 (Inverse)**.** Consider a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. Let matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ have the property that $\boldsymbol{AB} = \boldsymbol{I}_n = \boldsymbol{BA}$. $\boldsymbol{B}$ is called the *inverse* of $\boldsymbol{A}$ and denoted by $\boldsymbol{A}^{-1}$.

Matrices: inverse and transpose

Definition 2.3 (Inverse)**.** Consider a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. Let matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ have the property that $\boldsymbol{AB} = \boldsymbol{I}_n = \boldsymbol{BA}$. $\boldsymbol{B}$ is called the *inverse* of $\boldsymbol{A}$ and denoted by $\boldsymbol{A}^{-1}$.

Q: Is every matrix possesses an inverse?

Matrices: inverse and transpose

Definition 2.3 (Inverse)**.** Consider a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. Let matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ have the property that $\boldsymbol{AB} = \boldsymbol{I}_n = \boldsymbol{BA}$. $\boldsymbol{B}$ is called the *inverse* of $\boldsymbol{A}$ and denoted by $\boldsymbol{A}^{-1}$.

Q: Is every matrix possesses an inverse? No. Positive Definite Matrices are always invertible.

Matrices: inverse and transpose

Definition 2.3 (Inverse)**.** Consider a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. Let matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ have the property that $\boldsymbol{AB} = \boldsymbol{I}_n = \boldsymbol{BA}$. $\boldsymbol{B}$ is called the *inverse* of $\boldsymbol{A}$ and denoted by $\boldsymbol{A}^{-1}$.

Definition 2.4 (Transpose)**.** For $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ the matrix $\boldsymbol{B} \in \mathbb{R}^{n \times m}$ with $b_{ij} = a_{ji}$ is called the *transpose* of $\boldsymbol{A}$. We write $\boldsymbol{B} = \boldsymbol{A}^\top$.

In general, $\boldsymbol{A}^\top$ can be obtained by writing the columns of $\boldsymbol{A}$ as the rows of $\boldsymbol{A}^\top$. The following are important properties of inverses and transposes:

$$\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I} = \boldsymbol{A}^{-1}\boldsymbol{A} \tag{2.26}$$

$$(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1} \tag{2.27}$$

$$(\boldsymbol{A} + \boldsymbol{B})^{-1} \neq \boldsymbol{A}^{-1} + \boldsymbol{B}^{-1} \tag{2.28}$$

$$(\boldsymbol{A}^\top)^\top = \boldsymbol{A} \tag{2.29}$$

$$(\boldsymbol{A} + \boldsymbol{B})^\top = \boldsymbol{A}^\top + \boldsymbol{B}^\top \tag{2.30}$$

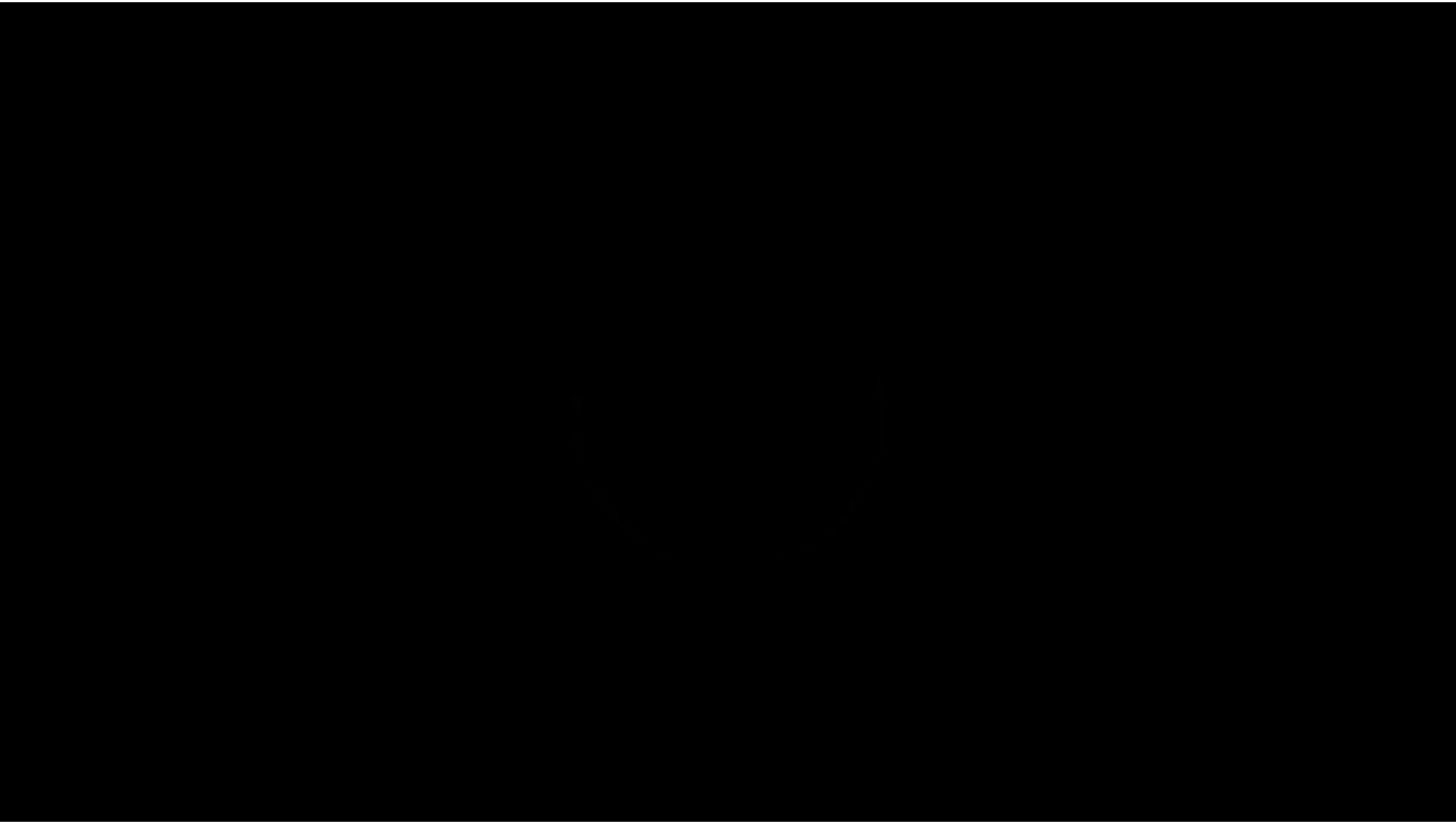
$$(\boldsymbol{AB})^\top = \boldsymbol{B}^\top \boldsymbol{A}^\top \tag{2.31}$$

Matrices: multiplication by a scalar

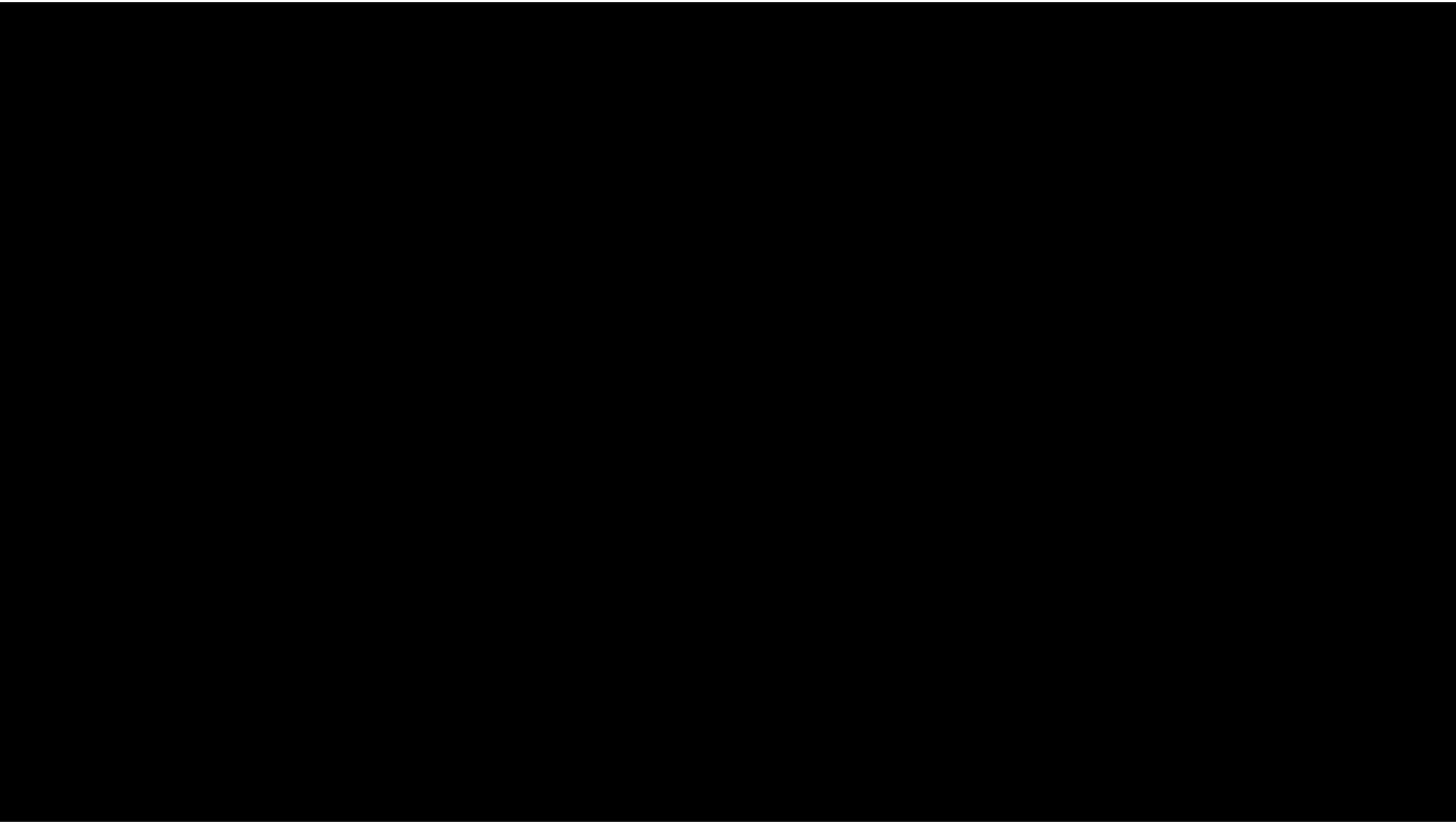
Let us look at what happens to matrices when they are multiplied by a scalar $\lambda \in \mathbb{R}$. Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and $\lambda \in \mathbb{R}$. Then $\lambda \boldsymbol{A} = \boldsymbol{K}$, $K_{ij} = \lambda a_{ij}$. Practically, λ scales each element of $\boldsymbol{A}$. For $\lambda, \psi \in \mathbb{R}$, the following holds:

- *Associativity:*
  $(\lambda\psi)\boldsymbol{C} = \lambda(\psi\boldsymbol{C}), \quad \boldsymbol{C} \in \mathbb{R}^{m \times n}$
- $\lambda(\boldsymbol{B}\boldsymbol{C}) = (\lambda\boldsymbol{B})\boldsymbol{C} = \boldsymbol{B}(\lambda\boldsymbol{C}) = (\boldsymbol{B}\boldsymbol{C})\lambda, \quad \boldsymbol{B} \in \mathbb{R}^{m \times n}, \boldsymbol{C} \in \mathbb{R}^{n \times k}$.
  Note that this allows us to move scalar values around.
- $(\lambda\boldsymbol{C})^\top = \boldsymbol{C}^\top\lambda^\top = \boldsymbol{C}^\top\lambda = \lambda\boldsymbol{C}^\top$ since $\lambda = \lambda^\top$ for all $\lambda \in \mathbb{R}$.
- *Distributivity:*
  $(\lambda + \psi)\boldsymbol{C} = \lambda\boldsymbol{C} + \psi\boldsymbol{C}, \quad \boldsymbol{C} \in \mathbb{R}^{m \times n}$
  $\lambda(\boldsymbol{B} + \boldsymbol{C}) = \lambda\boldsymbol{B} + \lambda\boldsymbol{C}, \quad \boldsymbol{B}, \boldsymbol{C} \in \mathbb{R}^{m \times n}$

Matrix of linear transformation

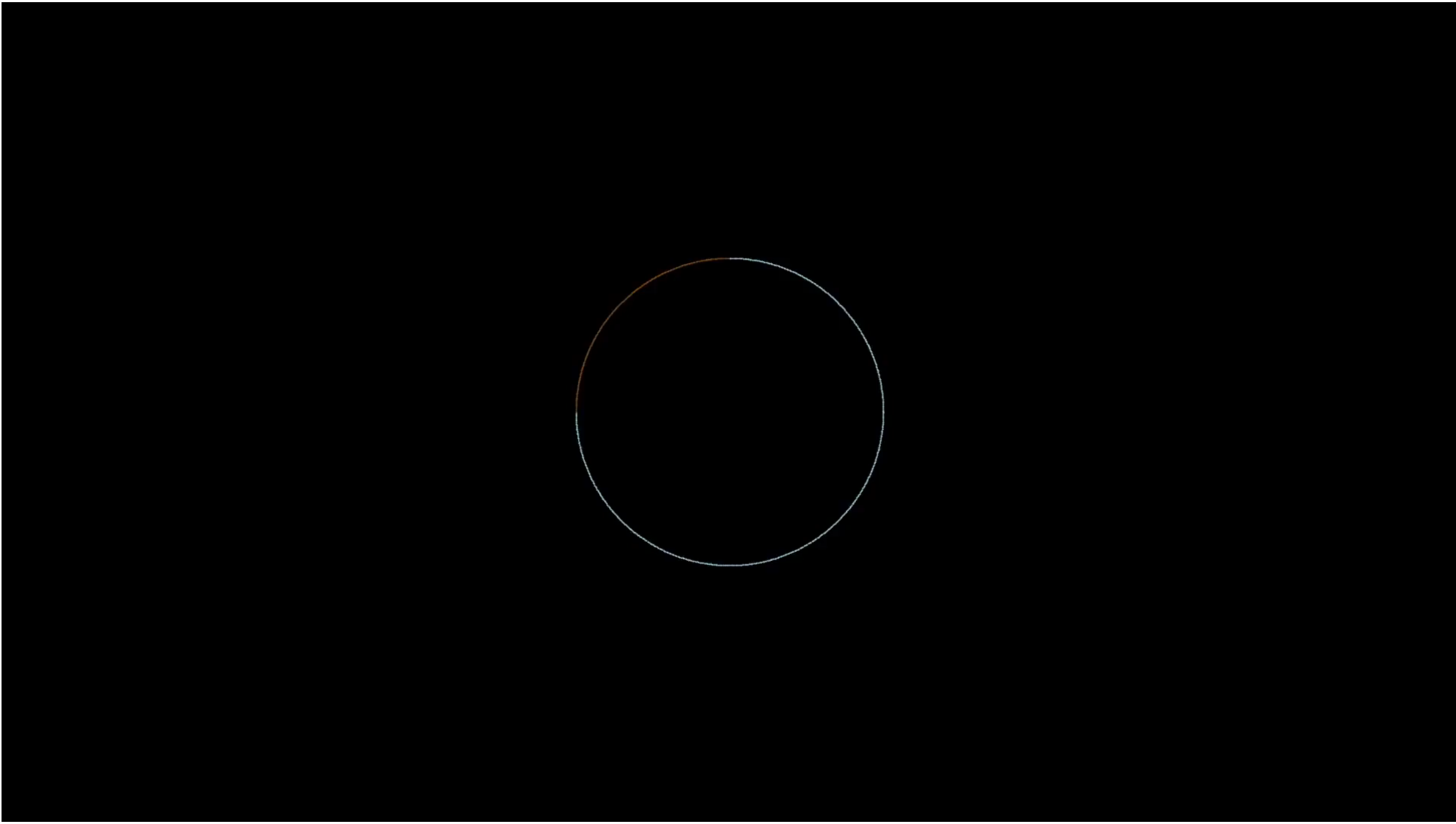


3d linear transformations



Matrix multiplication as composition

Eigenvectors and Eigenvalues



Google's PageRank

Google uses the eigenvector corresponding to the maximal eigenvalue of a matrix $\boldsymbol{A}$ to determine the rank of a page for search. The idea for the PageRank algorithm, developed at Stanford University by Larry Page and Sergey Brin in 1996, was that the importance of any web page can be approximated by the importance of pages that link to it. For this, they write down all web sites as a huge directed graph that shows which page links to which. PageRank computes the weight (importance) $x_i \geqslant 0$ of a web site a_i by counting the number of pages pointing to a_i. Moreover, PageRank takes into account the importance of the web sites that link to a_i. The navigation behavior of a user is then modeled by a transition matrix $\boldsymbol{A}$ of this graph that tells us with what (click) probability somebody will end up on a different web site. The matrix $\boldsymbol{A}$ has the property that for any initial rank/importance vector $\boldsymbol{x}$ of a web site the sequence $\boldsymbol{x}, \boldsymbol{A}\boldsymbol{x}, \boldsymbol{A}^2\boldsymbol{x}, \ldots$ converges to a vector $\boldsymbol{x}^*$. This vector is called the *PageRank* and satisfies $\boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{x}^*$, i.e., it is an eigenvector (with corresponding eigenvalue 1) of $\boldsymbol{A}$. After normalizing $\boldsymbol{x}^*$, such that $\|\boldsymbol{x}^*\| = 1$, we can interpret the entries as probabilities. More details and different perspectives on PageRank can be found in the original technical report (Page et al., 1999).

Low-rank matrix approximation

We considered the SVD as a way to factorize $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top \in \mathbb{R}^{m\times n}$ into the product of three matrices, where $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ are orthogonal and $\boldsymbol{\Sigma}$ contains the singular values on its main diagonal. Instead of doing the full SVD factorization, we will now investigate how the SVD allows us to represent a matrix $\boldsymbol{A}$ as a sum of simpler (low-rank) matrices $\boldsymbol{A}_i$, which lends itself to a matrix approximation scheme that is cheaper to compute than the full SVD.

We construct a rank-1 matrix $\boldsymbol{A}_i \in \mathbb{R}^{m\times n}$ as

$$\boldsymbol{A}_i := \boldsymbol{u}_i \boldsymbol{v}_i^\top , \tag{4.90}$$

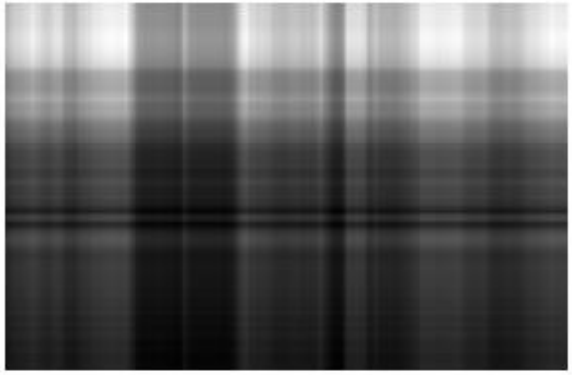
which is formed by the outer product of the ith orthogonal column vector of $\boldsymbol{U}$ and $\boldsymbol{V}$. Figure 4.11 shows an image of Stonehenge, which can be represented by a matrix $\boldsymbol{A} \in \mathbb{R}^{1432\times 1910}$, and some outer products $\boldsymbol{A}_i$, as defined in (4.90).

A matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ of rank r can be written as a sum of rank-1 matrices $\boldsymbol{A}_i$ so that

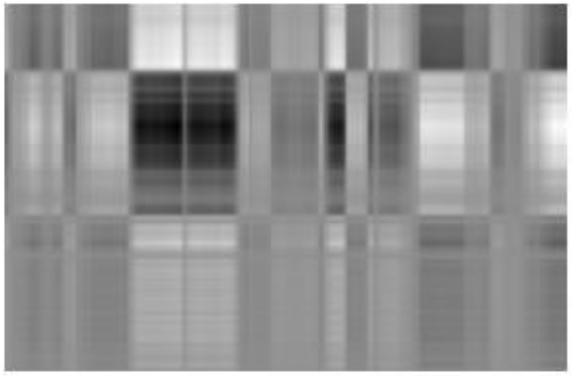
$$\boldsymbol{A} = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top = \sum_{i=1}^{r} \sigma_i \boldsymbol{A}_i , \tag{4.91}$$

Low-rank matrix approximation

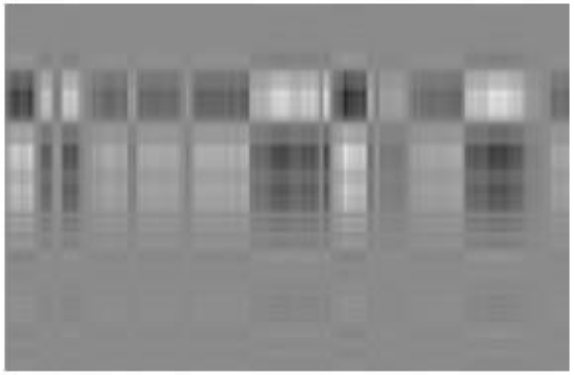
(a) Original image $\boldsymbol{A}$.



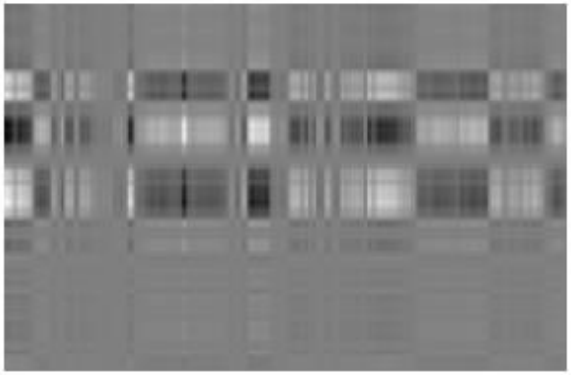
(b) $\boldsymbol{A}_1,\ \sigma_1 \approx 228,052.$



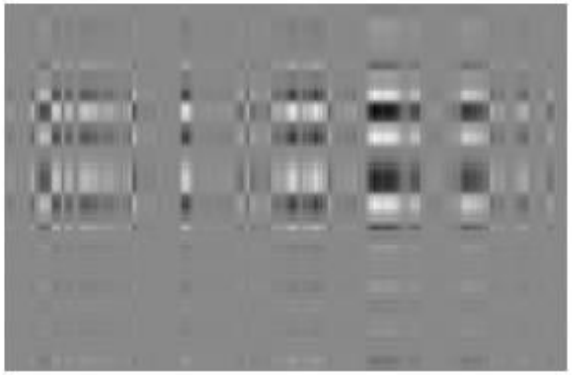
(c) $\boldsymbol{A}_2,\ \sigma_2 \approx 40,647.$



(d) $\boldsymbol{A}_3,\ \sigma_3 \approx 26,125.$



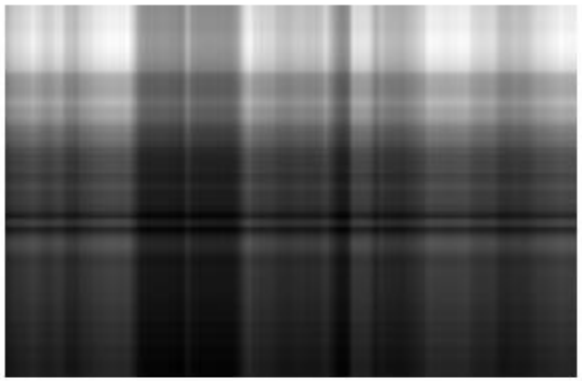
(e) $\boldsymbol{A}_4,\ \sigma_4 \approx 20,232.$



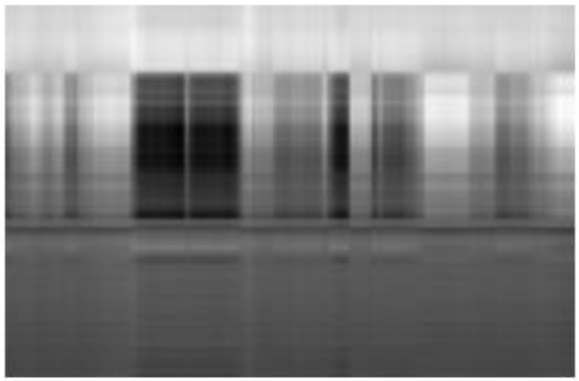
(f) $\boldsymbol{A}_5,\ \sigma_5 \approx 15,436.$

Low-rank matrix approximation

(a) Original image $\boldsymbol{A}$.



(b) Rank-1 approximation $\widehat{\boldsymbol{A}}(1)$.



(c) Rank-2 approximation $\widehat{\boldsymbol{A}}(2)$.

(d) Rank-3 approximation $\widehat{\boldsymbol{A}}(3)$.

(e) Rank-4 approximation $\widehat{\boldsymbol{A}}(4)$.

(f) Rank-5 approximation $\widehat{\boldsymbol{A}}(5)$.

Vector Calculus

Differentiation Rules

In the following, we briefly state basic differentiation rules, where we denote the derivative of f by f'.

$$\text{Product rule:} \quad (f(x)g(x))' = f'(x)g(x) + f(x)g'(x) \tag{5.29}$$

$$\text{Quotient rule:} \quad \left(\frac{f(x)}{g(x)}\right)' = \frac{f'(x)g(x) - f(x)g'(x)}{(g(x))^2} \tag{5.30}$$

$$\text{Sum rule:} \quad (f(x) + g(x))' = f'(x) + g'(x) \tag{5.31}$$

$$\text{Chain rule:} \quad \big(g(f(x))\big)' = (g \circ f)'(x) = g'(f(x))f'(x) \tag{5.32}$$

Here, $g \circ f$ denotes function composition $x \mapsto f(x) \mapsto g(f(x))$.

Partial Differentiation and Gradients

Definition 5.5 (Partial Derivative)**.** For a function $f : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{x} \mapsto f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^n$ of n variables $x_1, \ldots, x_n$ we define the *partial derivatives* as

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= \lim_{h \to 0} \frac{f(x_1 + h, x_2, \ldots, x_n) - f(\boldsymbol{x})}{h} \\ &\vdots \\ \frac{\partial f}{\partial x_n} &= \lim_{h \to 0} \frac{f(x_1, \ldots, x_{n-1}, x_n + h) - f(\boldsymbol{x})}{h} \end{aligned} \tag{5.39}$$

and collect them in the row vector

$$\nabla_{\boldsymbol{x}} f = \operatorname{grad} f = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f(\boldsymbol{x})}{\partial x_1} & \frac{\partial f(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial f(\boldsymbol{x})}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{1 \times n}, \tag{5.40}$$

Gradients of Vector-Valued Functions

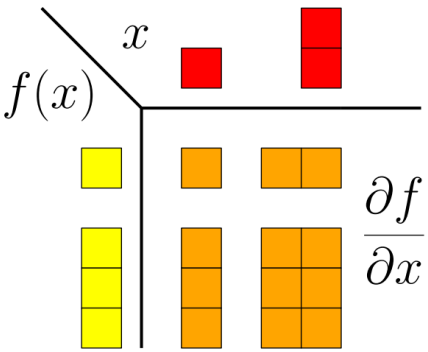
Definition 5.6 (Jacobian). The collection of all first-order partial derivatives of a vector-valued function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ is called the *Jacobian*. The Jacobian $\boldsymbol{J}$ is an $m \times n$ matrix, which we define and arrange as follows:

$$\boldsymbol{J} = \nabla_{\boldsymbol{x}} \boldsymbol{f} = \frac{\mathrm{d}\boldsymbol{f}(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_n} \end{bmatrix} \tag{5.57}$$

$$= \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_m(\boldsymbol{x})}{\partial x_n} \end{bmatrix}, \tag{5.58}$$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad J(i,j) = \frac{\partial f_i}{\partial x_j}. \tag{5.59}$$

Figure 5.6 Dimensionality of (partial) derivatives.



Chain Rule for vector differentiation

向量求导的链式法则：

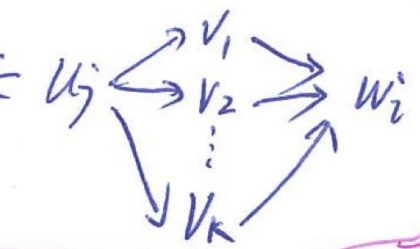
若多个向量的依赖关系为 $u \to v \to w$.

则 $\frac{\partial w}{\partial u} = \frac{\partial w}{\partial v} \cdot \frac{\partial v}{\partial u}$

u: $m\times 1$, v: $n\times 1$, w: $l\times 1$

$\frac{\partial w}{\partial u}$: $l\times m$, $\frac{\partial w}{\partial v}$: $l\times n$, $\frac{\partial v}{\partial u}$: $n\times m$.

证明：$\frac{\partial w_i}{\partial u_j} = \sum_k \frac{\partial w_i}{\partial v_k} \cdot \frac{\partial v_k}{\partial u_j}$



该法则可推广到多层中间变量的情形

A simplest example

Example 5.9 (Gradient of a Vector-Valued Function)
We are given

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}\,, \qquad \boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}^M, \quad \boldsymbol{A} \in \mathbb{R}^{M\times N}, \quad \boldsymbol{x} \in \mathbb{R}^N\,.$$

To compute the gradient $\mathrm{d}\boldsymbol{f}/\mathrm{d}\boldsymbol{x}$ we first determine the dimension of $\mathrm{d}\boldsymbol{f}/\mathrm{d}\boldsymbol{x}$: Since $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^M$, it follows that $\mathrm{d}\boldsymbol{f}/\mathrm{d}\boldsymbol{x} \in \mathbb{R}^{M\times N}$. Second, to compute the gradient we determine the partial derivatives of f with respect to every x_j:

$$f_i(\boldsymbol{x}) = \sum_{j=1}^{N} A_{ij}x_j \implies \frac{\partial f_i}{\partial x_j} = A_{ij} \tag{5.67}$$

We collect the partial derivatives in the Jacobian and obtain the gradient

$$\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & & \vdots \\ \frac{\partial f_M}{\partial x_1} & \cdots & \frac{\partial f_M}{\partial x_N} \end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & & \vdots \\ A_{M1} & \cdots & A_{MN} \end{bmatrix} = \boldsymbol{A} \in \mathbb{R}^{M\times N}\,. \tag{5.68}$$

More cases

① $\nabla Ax = A$，特别地，$\nabla x = I$

证：$y = Ax \Rightarrow \dfrac{\partial y_i}{\partial x_j} = \dfrac{\partial \sum_j A_{ij} x_j}{\partial x_j} = A_{ij}$

② $\nabla(a^T x) = a^T$

证：$\dfrac{\partial a^T x}{\partial x_i} = \dfrac{\partial \sum_j a_j x_j}{\partial x_i} = \dfrac{\partial a_i x_i}{\partial x_i} = a_i$

③ $\nabla \|x\|_2^2 = \nabla(x^T x) = 2x$

证：$\dfrac{\partial \|x\|_2^2}{\partial x_i} = \dfrac{\partial \sum_i x_i^2}{\partial x_i} = \dfrac{\partial x_i^2}{\partial x_i} = 2x_i$

More cases

④ $\nabla(x^TAx) = x^T(A+A^T)$

证：

$f(x) = x^TAx$

↓ 添加下标

$f = x_1^TAx_2$

↓ 分别求导

$\frac{\partial f}{\partial x_1} = x_2^TA^T$，$\frac{\partial f}{\partial x_2} = x_1^TA$

↓ 去掉下标后相加

$\frac{\partial f}{\partial x} = x^T(A+A^T)$

More cases

$l\times m \quad l\times n \quad n\times m \quad l\times n \quad n\times m$

⑤ $\nabla(u^T v) = v^T(\nabla_x u) + u^T(\nabla_x v)$

注：利用变量多次出现的求导法则（x在u，v中出现）.

⑥ $\nabla_x(\alpha(x)f(x)) = f(x)\cdot\nabla_x\alpha(x) + \alpha(x)\cdot\nabla_x f(x)$

$m\times n \quad m\times 1 \quad 1\times n \quad 1\times 1 \quad m\times n$

让 $\dfrac{\partial \alpha f_i}{\partial x_j} = f_i\cdot\dfrac{\partial \alpha}{\partial x_j} + \alpha\cdot\dfrac{\partial f_i}{\partial x_j}$.

More cases

⑦ $\nabla tr(A^TX) = \nabla tr(AX^T) = A$

证 $\frac{\partial tr(A^TX)}{\partial X_{ij}} = \frac{\partial \sum_{ij}(A_{ij} \cdot X_{ij})}{\partial X_{ij}} = A_{ij}$

⑧ $\nabla tr(XAX^TB) = B^TXA^T + BXA$

证 利用变量多次出现的求导法则.

⑨ $\nabla a^TXb = ab^T$

证 $LHS = \nabla tr(a^TXb) = \nabla tr(Xba^T) = ab^T = RHS$

More cases

⑩ $\nabla_X a^T X^T X a = 2Xaa^T$ 直接用⑧证明.

⑪ $\nabla_X (Xa-b)^T (Xa-b) = 2(Xa-b)a^T$ 既然后用⑩证明

⑫ $\nabla_X \|XA^T - B\|_F^2 = 2(XA^T - B)A$

注意 $\|A\|_F^2 = \sum_{ij} A_{ij}^2 = tr(A^T A)$

⑬行列式：$\nabla_X |X| = |X| (X^{-1})^T$

More cases

⑭ 设 $y=f(u)$，$u=g(X)$，则 $\frac{\partial y}{\partial X}=\frac{\partial y}{\partial u}\cdot\frac{\partial u}{\partial X}$

实数 实数 矩阵 $m\times n$ 1×1 $m\times n$

⑮ 设 $f(Y):R^{m\times p}\to R$，则

$\nabla_X f(AX+B)=A^T\nabla_{Y=AX+B}f$

$m\times n$ $n\times p$ $m\times p$

$\nabla_X f(XC+D)=(\nabla_{Y=XC+D}f)\cdot C^T$

⑯ $\nabla_x f(Ax+b)=A^T\nabla_{y=Ax+b}f$

⑮的向量版本

Useful Tricks

Trick: turn all to vector/matrix operations.

$$\begin{cases} \sum_i u_i v_i = u^T v \\ \sum_{ij} A_{ij} B_{ij} = tr(AB^T) \\ \sum_i x_i^2 = x^T x \\ \|A\|_F^2 = tr(AA^T) \end{cases}$$

Useful Identities

$$\frac{\partial}{\partial \boldsymbol{X}} \boldsymbol{f}(\boldsymbol{X})^\top = \left(\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\right)^\top \tag{5.99}$$

$$\frac{\partial}{\partial \boldsymbol{X}} \mathrm{tr}(\boldsymbol{f}(\boldsymbol{X})) = \mathrm{tr}\left(\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\right) \tag{5.100}$$

$$\frac{\partial}{\partial \boldsymbol{X}} \det(\boldsymbol{f}(\boldsymbol{X})) = \det(\boldsymbol{f}(\boldsymbol{X}))\mathrm{tr}\left(\boldsymbol{f}(\boldsymbol{X})^{-1}\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\right) \tag{5.101}$$

$$\frac{\partial}{\partial \boldsymbol{X}} \boldsymbol{f}(\boldsymbol{X})^{-1} = -\boldsymbol{f}(\boldsymbol{X})^{-1}\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\boldsymbol{f}(\boldsymbol{X})^{-1} \tag{5.102}$$

$$\frac{\partial \boldsymbol{a}^\top \boldsymbol{X}^{-1}\boldsymbol{b}}{\partial \boldsymbol{X}} = -(\boldsymbol{X}^{-1})^\top \boldsymbol{a}\boldsymbol{b}^\top (\boldsymbol{X}^{-1})^\top \tag{5.103}$$

$$\frac{\partial \boldsymbol{x}^\top \boldsymbol{a}}{\partial \boldsymbol{x}} = \boldsymbol{a}^\top \tag{5.104}$$

$$\frac{\partial \boldsymbol{a}^\top \boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{a}^\top \tag{5.105}$$

$$\frac{\partial \boldsymbol{a}^\top \boldsymbol{X}\boldsymbol{b}}{\partial \boldsymbol{X}} = \boldsymbol{a}\boldsymbol{b}^\top \tag{5.106}$$

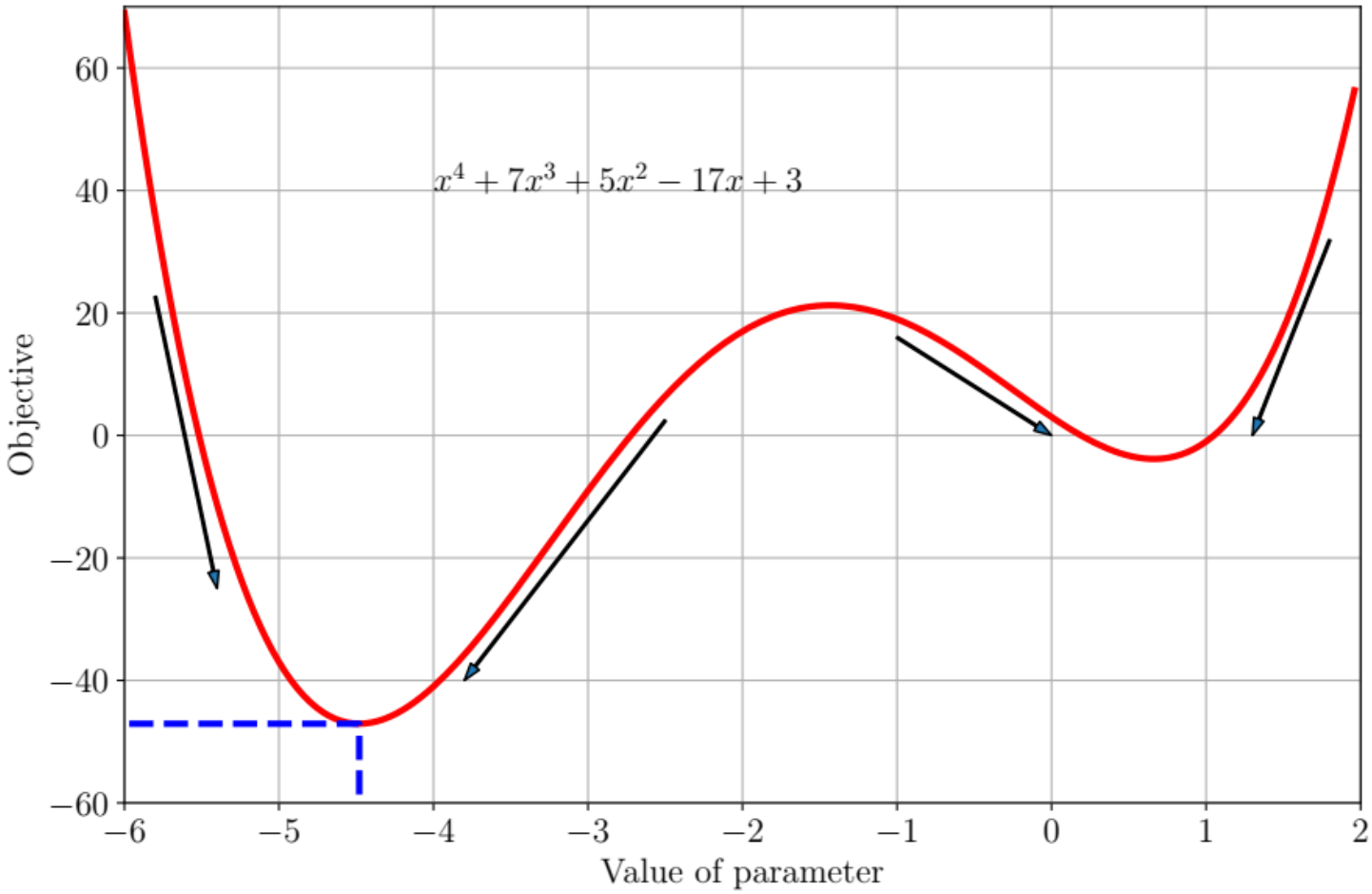
$$\frac{\partial \boldsymbol{x}^\top \boldsymbol{B}\boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{x}^\top(\boldsymbol{B} + \boldsymbol{B}^\top) \tag{5.107}$$

$$\frac{\partial}{\partial \boldsymbol{s}}(\boldsymbol{x} - \boldsymbol{A}\boldsymbol{s})^\top \boldsymbol{W}(\boldsymbol{x} - \boldsymbol{A}\boldsymbol{s}) = -2(\boldsymbol{x} - \boldsymbol{A}\boldsymbol{s})^\top \boldsymbol{W}\boldsymbol{A} \quad \text{for symmetric } \boldsymbol{W} \tag{5.108}$$

Optimization

Machine Learning as Optimization

- **The process of machine learning is to find parameters that minimize objective**



Gradient Descent

We now consider the problem of solving for the minimum of a real-valued function

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \,, \tag{7.4}$$

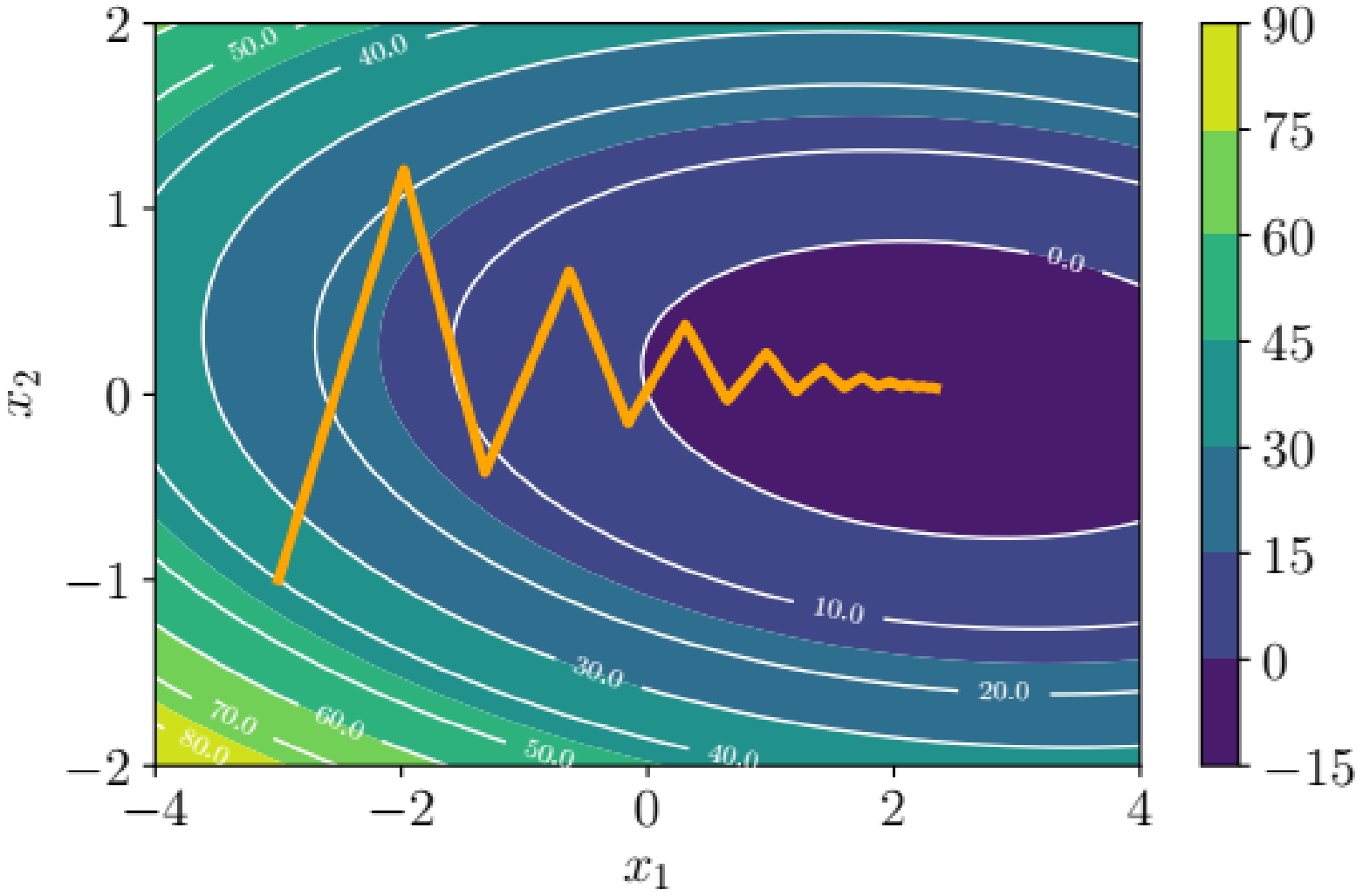
where $f : \mathbb{R}^d \to \mathbb{R}$ is an objective function that captures the machine learning problem at hand. We assume that our function f is differentiable, and we are unable to analytically find a solution in closed form.

Gradient descent is a first-order optimization algorithm. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient of the function at the current point.

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \gamma_i((\nabla f)(\boldsymbol{x}_i))^\top \,. \tag{7.6}$$

For suitable step-size γ_i, the sequence $f(\boldsymbol{x}_0) \geqslant f(\boldsymbol{x}_1) \geqslant \ldots$ converges to a local minimum.

Gradient Descent



Stochastic Gradient Descent

In machine learning, given $n = 1, \ldots, N$ data points, we often consider objective functions that are the sum of the losses L_n incurred by each example n. In mathematical notation, we have the form

$$L(\boldsymbol{\theta}) = \sum_{n=1}^{N} L_n(\boldsymbol{\theta}) \,, \tag{7.13}$$

Standard gradient descent, as introduced previously, is a "batch" optimization method, i.e., optimization is performed using the full training set by updating the vector of parameters according to

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \gamma_i (\nabla L(\boldsymbol{\theta}_i))^\top = \boldsymbol{\theta}_i - \gamma_i \sum_{n=1}^{N} (\nabla L_n(\boldsymbol{\theta}_i))^\top \tag{7.15}$$

Consider the term $\sum_{n=1}^{N} (\nabla L_n(\boldsymbol{\theta}_i))$ in (7.15). We can reduce the amount of computation by taking a sum over a smaller set of L_n. In contrast to batch gradient descent, which uses all L_n for $n = 1, \ldots, N$, we randomly choose a subset of L_n for mini-batch gradient descent. In the extreme case, we randomly select only a single L_n to estimate the gradient. The

Probabilities

Sum Rule and Product Rule

The first rule, the *sum rule*, states that

$$p(\boldsymbol{x}) = \begin{cases} \sum_{\boldsymbol{y} \in \mathcal{Y}} p(\boldsymbol{x}, \boldsymbol{y}) & \text{if } \boldsymbol{y} \text{ is discrete} \\ \int_{\mathcal{Y}} p(\boldsymbol{x}, \boldsymbol{y}) \mathrm{d}\boldsymbol{y} & \text{if } \boldsymbol{y} \text{ is continuous} \end{cases}, \tag{6.20}$$

where $\mathcal{Y}$ are the states of the target space of random variable Y. This

The second rule, known as the *product rule*, relates the joint distribution to the conditional distribution via

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{y} \,|\, \boldsymbol{x}) p(\boldsymbol{x}) \,. \tag{6.22}$$

Bayes' Rule

In machine learning and Bayesian statistics, we are often interested in making inferences of unobserved (latent) random variables given that we have observed other random variables. Let us assume we have some prior knowledge $p(\boldsymbol{x})$ about an unobserved random variable $\boldsymbol{x}$ and some relationship $p(\boldsymbol{y} \mid \boldsymbol{x})$ between $\boldsymbol{x}$ and a second random variable $\boldsymbol{y}$, which we can observe. If we observe $\boldsymbol{y}$, we can use Bayes' theorem to draw some conclusions about $\boldsymbol{x}$ given the observed values of $\boldsymbol{y}$. *Bayes' theorem* (also *Bayes' rule* or *Bayes' law*)

$$\underbrace{p(\boldsymbol{x} \mid \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y} \mid \boldsymbol{x})}^{\text{likelihood}} \overbrace{p(\boldsymbol{x})}^{\text{prior}}}{\underbrace{p(\boldsymbol{y})}_{\text{evidence}}} \tag{6.23}$$

Bayes' Rule

In (6.23), $p(\boldsymbol{x})$ is the *prior*, which encapsulates our subjective prior knowledge of the unobserved (latent) variable $\boldsymbol{x}$ before observing any data. We can choose any prior that makes sense to us, but it is critical to ensure that the prior has a nonzero pdf (or pmf) on all plausible $\boldsymbol{x}$, even if they are very rare.

The *likelihood* $p(\boldsymbol{y} \mid \boldsymbol{x})$ describes how $\boldsymbol{x}$ and $\boldsymbol{y}$ are related, and in the case of discrete probability distributions, it is the probability of the data $\boldsymbol{y}$ if we were to know the latent variable $\boldsymbol{x}$. Note that the likelihood is not a distribution in $\boldsymbol{x}$, but only in $\boldsymbol{y}$. We call $p(\boldsymbol{y} \mid \boldsymbol{x})$ either the "likelihood of $\boldsymbol{x}$ (given $\boldsymbol{y}$)" or the "probability of $\boldsymbol{y}$ given $\boldsymbol{x}$" but never the likelihood of $\boldsymbol{y}$ (MacKay, 2003).

The *posterior* $p(\boldsymbol{x} \mid \boldsymbol{y})$ is the quantity of interest in Bayesian statistics because it expresses exactly what we are interested in, i.e., what we know about $\boldsymbol{x}$ after having observed $\boldsymbol{y}$.

Expected Value

Definition 6.3 (Expected Value)**.** The *expected value* of a function $g : \mathbb{R} \to \mathbb{R}$ of a univariate continuous random variable $X \sim p(x)$ is given by

$$\mathbb{E}_X[g(x)] = \int_{\mathcal{X}} g(x)p(x)\mathrm{d}x\,. \tag{6.28}$$

Correspondingly, the expected value of a function g of a discrete random variable $X \sim p(x)$ is given by

$$\mathbb{E}_X[g(x)] = \sum_{x\in\mathcal{X}} g(x)p(x)\,, \tag{6.29}$$

where $\mathcal{X}$ is the set of possible outcomes (the target space) of the random variable X.

Covariance and Variance

Definition 6.6 (Covariance (Multivariate)). If we consider two multivariate random variables X and Y with states $\boldsymbol{x} \in \mathbb{R}^D$ and $\boldsymbol{y} \in \mathbb{R}^E$ respectively, the *covariance* between X and Y is defined as

$$\operatorname{Cov}[\boldsymbol{x}, \boldsymbol{y}] = \mathbb{E}[\boldsymbol{x}\boldsymbol{y}^\top] - \mathbb{E}[\boldsymbol{x}]\mathbb{E}[\boldsymbol{y}]^\top = \operatorname{Cov}[\boldsymbol{y}, \boldsymbol{x}]^\top \in \mathbb{R}^{D \times E}. \tag{6.37}$$

Definition 6.7 (Variance). The *variance* of a random variable X with states $\boldsymbol{x} \in \mathbb{R}^D$ and a mean vector $\boldsymbol{\mu} \in \mathbb{R}^D$ is defined as

$$\mathbb{V}_X[\boldsymbol{x}] = \operatorname{Cov}_X[\boldsymbol{x}, \boldsymbol{x}] \tag{6.38a}$$

$$= \mathbb{E}_X[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top] = \mathbb{E}_X[\boldsymbol{x}\boldsymbol{x}^\top] - \mathbb{E}_X[\boldsymbol{x}]\mathbb{E}_X[\boldsymbol{x}]^\top \tag{6.38b}$$

$$= \begin{bmatrix} \operatorname{Cov}[x_1, x_1] & \operatorname{Cov}[x_1, x_2] & \dots & \operatorname{Cov}[x_1, x_D] \\ \operatorname{Cov}[x_2, x_1] & \operatorname{Cov}[x_2, x_2] & \dots & \operatorname{Cov}[x_2, x_D] \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{Cov}[x_D, x_1] & \dots & \dots & \operatorname{Cov}[x_D, x_D] \end{bmatrix}. \tag{6.38c}$$

Empirical Mean and Covariance

Definition 6.9 (Empirical Mean and Covariance)**.** The *empirical mean* vector is the arithmetic average of the observations for each variable, and it is defined as

$$\bar{\boldsymbol{x}} := \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n \,, \tag{6.41}$$

where $\boldsymbol{x}_n \in \mathbb{R}^D$.

Similar to the empirical mean, the *empirical covariance* matrix is a $D \times D$ matrix

$$\boldsymbol{\Sigma} := \frac{1}{N}\sum_{n=1}^{N} (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^\top . \tag{6.42}$$

Sums of Random Variables

Consider two random variables X, Y with states $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^D$. Then:

$$\mathbb{E}[\boldsymbol{x} + \boldsymbol{y}] = \mathbb{E}[\boldsymbol{x}] + \mathbb{E}[\boldsymbol{y}] \tag{6.46}$$

$$\mathbb{E}[\boldsymbol{x} - \boldsymbol{y}] = \mathbb{E}[\boldsymbol{x}] - \mathbb{E}[\boldsymbol{y}] \tag{6.47}$$

$$\mathbb{V}[\boldsymbol{x} + \boldsymbol{y}] = \mathbb{V}[\boldsymbol{x}] + \mathbb{V}[\boldsymbol{y}] + \mathrm{Cov}[\boldsymbol{x}, \boldsymbol{y}] + \mathrm{Cov}[\boldsymbol{y}, \boldsymbol{x}] \tag{6.48}$$

$$\mathbb{V}[\boldsymbol{x} - \boldsymbol{y}] = \mathbb{V}[\boldsymbol{x}] + \mathbb{V}[\boldsymbol{y}] - \mathrm{Cov}[\boldsymbol{x}, \boldsymbol{y}] - \mathrm{Cov}[\boldsymbol{y}, \boldsymbol{x}]\,. \tag{6.49}$$

Transformations of Random Variables

Mean and (co)variance exhibit some useful properties when it comes to affine transformation of random variables. Consider a random variable X with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ and a (deterministic) affine transformation $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$ of $\boldsymbol{x}$. Then $\boldsymbol{y}$ is itself a random variable whose mean vector and covariance matrix are given by

$$\mathbb{E}_Y[\boldsymbol{y}] = \mathbb{E}_X[\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}] = \boldsymbol{A}\mathbb{E}_X[\boldsymbol{x}] + \boldsymbol{b} = \boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b}\,, \tag{6.50}$$

$$\mathbb{V}_Y[\boldsymbol{y}] = \mathbb{V}_X[\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}] = \mathbb{V}_X[\boldsymbol{A}\boldsymbol{x}] = \boldsymbol{A}\mathbb{V}_X[\boldsymbol{x}]\boldsymbol{A}^\top = \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^\top\,, \tag{6.51}$$

respectively. Furthermore,

$$\begin{aligned}
\operatorname{Cov}[\boldsymbol{x}, \boldsymbol{y}] &= \mathbb{E}[\boldsymbol{x}(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b})^\top] - \mathbb{E}[\boldsymbol{x}]\mathbb{E}[\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}]^\top && \text{(6.52a)}\\
&= \mathbb{E}[\boldsymbol{x}]\boldsymbol{b}^\top + \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top]\boldsymbol{A}^\top - \boldsymbol{\mu}\boldsymbol{b}^\top - \boldsymbol{\mu}\boldsymbol{\mu}^\top\boldsymbol{A}^\top && \text{(6.52b)}\\
&= \boldsymbol{\mu}\boldsymbol{b}^\top - \boldsymbol{\mu}\boldsymbol{b}^\top + \left(\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top] - \boldsymbol{\mu}\boldsymbol{\mu}^\top\right)\boldsymbol{A}^\top && \text{(6.52c)}\\
&\overset{(6.38b)}{=} \boldsymbol{\Sigma}\boldsymbol{A}^\top\,, && \text{(6.52d)}
\end{aligned}$$

where $\boldsymbol{\Sigma} = \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top] - \boldsymbol{\mu}\boldsymbol{\mu}^\top$ is the covariance of X.

Statistical Independence

Definition 6.10 (Independence)**.** Two random variables X, Y are *statistically independent* if and only if

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x})p(\boldsymbol{y})\,. \tag{6.53}$$

Intuitively, two random variables X and Y are independent if the value of $\boldsymbol{y}$ (once known) does not add any additional information about $\boldsymbol{x}$ (and vice versa). If X, Y are (statistically) independent, then

- $p(\boldsymbol{y} \mid \boldsymbol{x}) = p(\boldsymbol{y})$
- $p(\boldsymbol{x} \mid \boldsymbol{y}) = p(\boldsymbol{x})$
- $\mathbb{V}_{X,Y}[\boldsymbol{x} + \boldsymbol{y}] = \mathbb{V}_X[\boldsymbol{x}] + \mathbb{V}_Y[\boldsymbol{y}]$
- $\mathrm{Cov}_{X,Y}[\boldsymbol{x}, \boldsymbol{y}] = \mathbf{0}$

Definition 6.11 (Conditional Independence)**.** Two random variables X and Y are *conditionally independent* given Z if and only if

$$p(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{y} \mid \boldsymbol{z}) \quad \text{for all} \quad \boldsymbol{z} \in \mathcal{Z}\,, \tag{6.55}$$

where $\mathcal{Z}$ is the set of states of random variable Z. We write $X \perp\!\!\!\perp Y \mid Z$ to denote that X is conditionally independent of Y given Z.

Gaussian Distribution

For a univariate random variable, the Gaussian distribution has a density that is given by

$$p(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right). \tag{6.62}$$

The *multivariate Gaussian distribution* is fully characterized by a *mean vector* $\boldsymbol{\mu}$ and a *covariance matrix* $\boldsymbol{\Sigma}$ and defined as

$$p(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{D}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right), \tag{6.63}$$

where $\boldsymbol{x} \in \mathbb{R}^D$. We write $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ or $X \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Fig-

Gaussians are widely used in statistical estimation and machine learning as they have closed-form expressions for marginal and conditional distributions.