컴퓨터 비전 초격차: 이론/실습

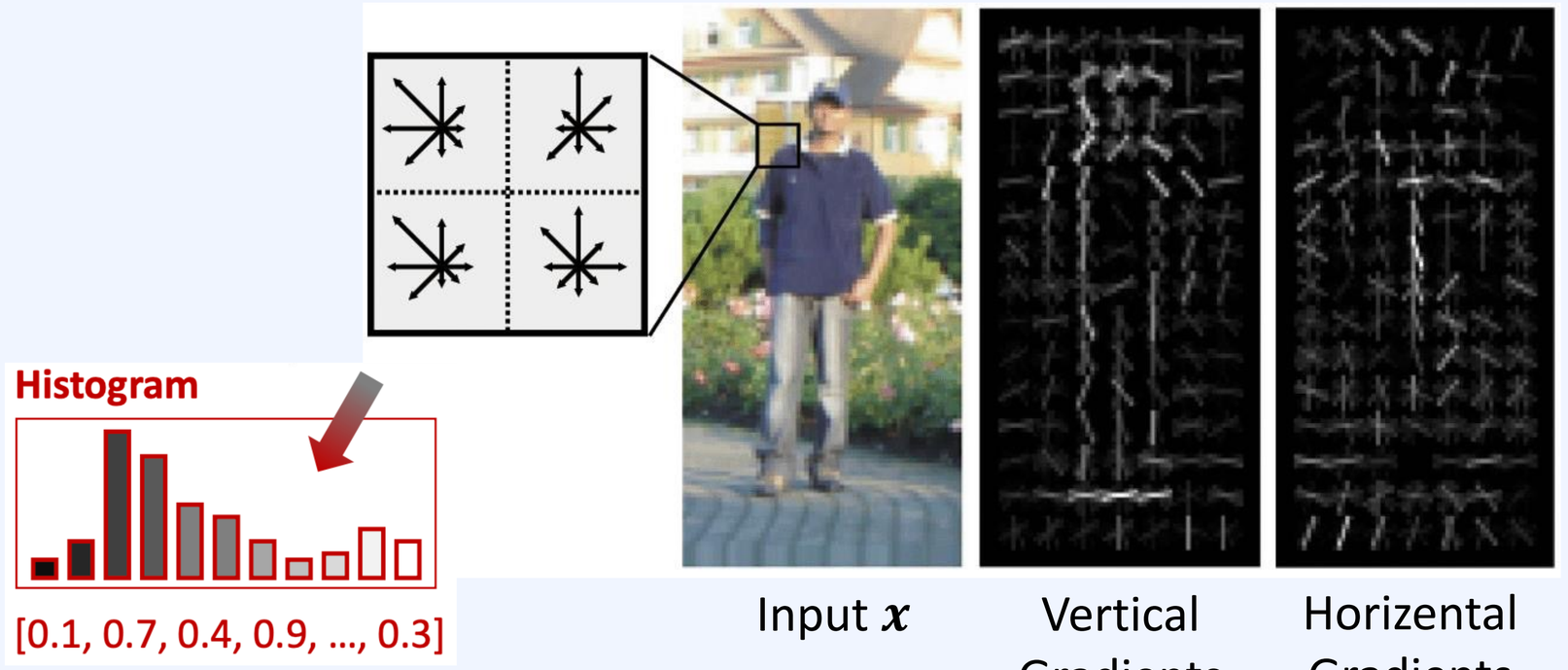
2 Preliminary – Linear algebra

Index

Most of the slides are inspired by POSTECH AIGS549 class (Prof. Suha Kwak)

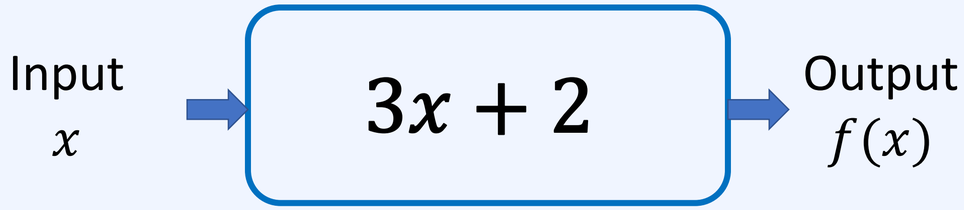
Linear Algebra in Computer Vision

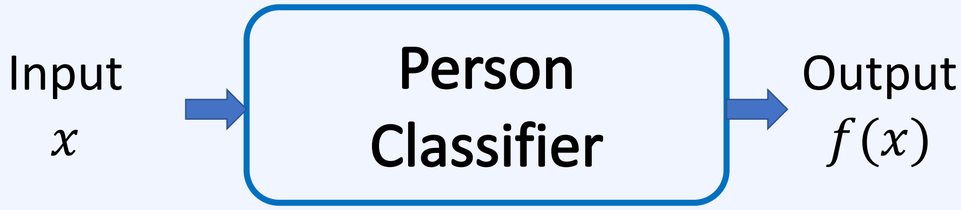
- Mathematics in vector space
- E.g., Image understanding = Image description (Vector) + Decision Making

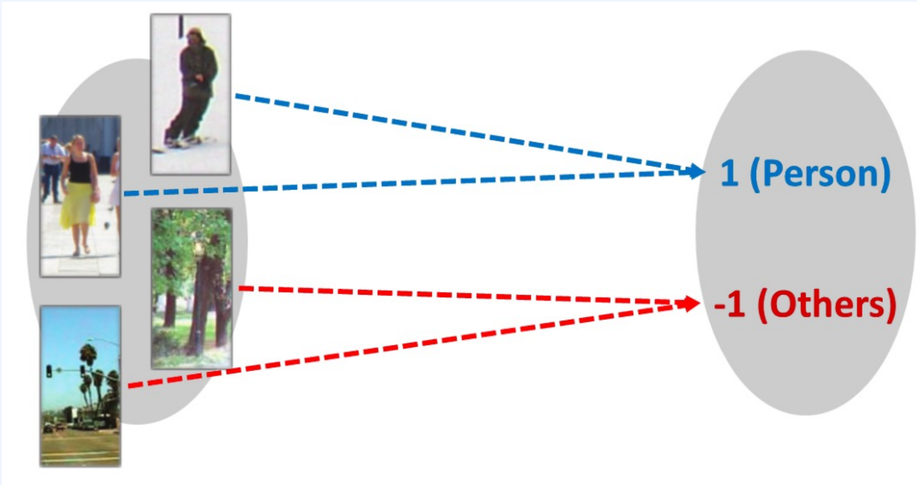


Linear Algebra in Computer Vision

- Mathematics in vector space
- E.g., Image understanding = Image description (Vector) + Decision Making (Function)

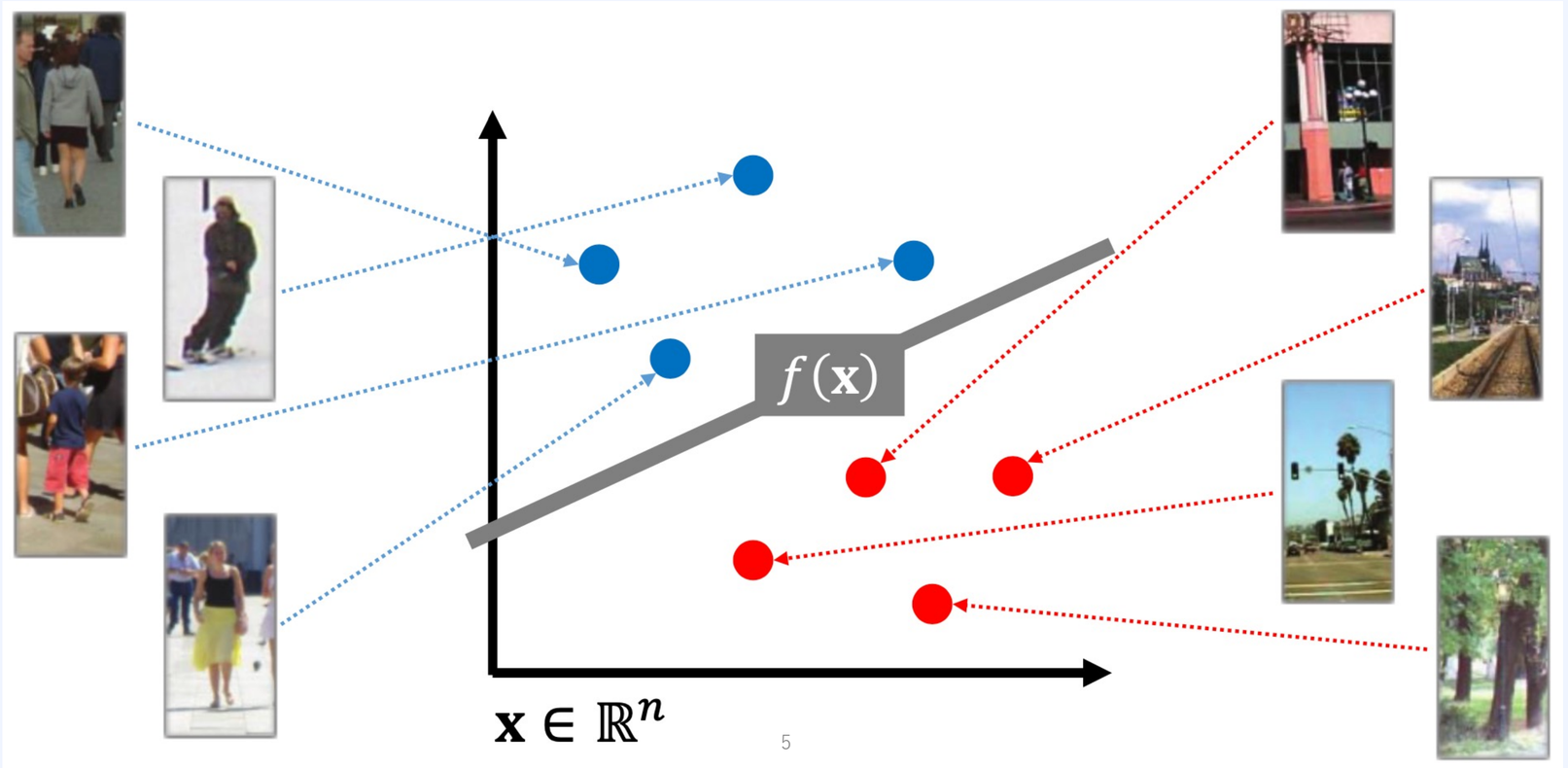






Linear Algebra in Computer Vision

- Mathematics in vector space
- E.g., Image understanding = Image description (Vector) + Decision Making (Function)



Linear Algebra in Computer Vision

- Mathematics in vector space
- Frequently used in
  - Image description
  - Computation of similarities and distances
  - Finding algebraic solutions
  - Transformations
  - Optimization

Vector

- Geometric object that has both a magnitude and direction
- Notation

$$\boldsymbol{x} \in \mathbb{R}^n, \qquad \boldsymbol{x} \in \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

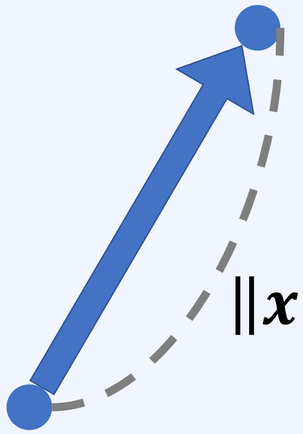
- Transpose

$$\boldsymbol{x}^\top = [x_1 \quad x_2 \quad \cdots \quad x_n]$$

- Magnitude

$$\|\boldsymbol{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

$$\boldsymbol{x} = [x_1 \quad x_2 \quad \cdots \quad x_n]^\top$$



$$\boldsymbol{0} = [0 \quad 0 \quad \ldots \quad 0]^\top$$

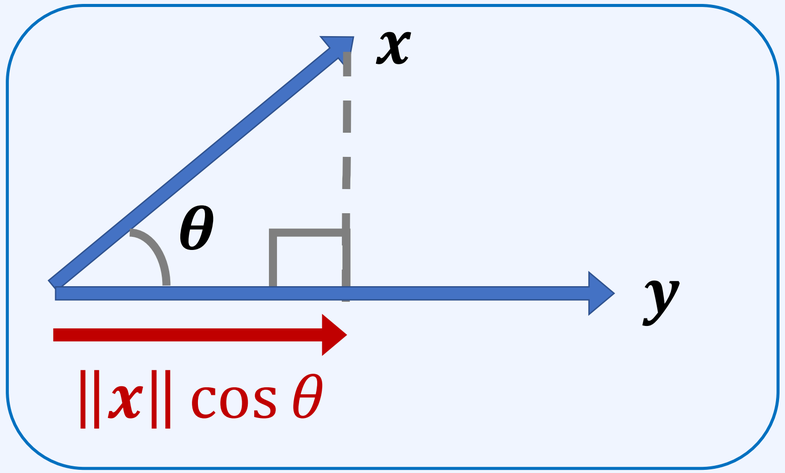
Vector Operations

- Inner product (*a.k.a.*, dot product)

$$\boldsymbol{x} = [x_1 \quad x_2 \quad \cdots \quad x_n]^\top \quad \boldsymbol{y} = [y_1 \quad y_2 \quad \cdots \quad y_n]^\top$$

$$\boldsymbol{x} \cdot \boldsymbol{y} = \|\boldsymbol{x}\|\|\boldsymbol{y}\| \cos\theta$$

$$= \sum_{i=1}^{n} x_i y_i = x_1 y_1 + \cdots + x_n y_n = \boldsymbol{x}^\top \boldsymbol{y}$$



- Properties of inner product
  - Symmetry: $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{y} \cdot \boldsymbol{x}$
  - Distributiveness: $(\boldsymbol{x} + \boldsymbol{y}) \cdot \boldsymbol{z} = \boldsymbol{x} \cdot \boldsymbol{z} + \boldsymbol{y} \cdot \boldsymbol{z}$
  - Linearity: $(\boldsymbol{\lambda x}) \cdot \boldsymbol{y} = \boldsymbol{x} \cdot (\boldsymbol{\lambda y}) = \boldsymbol{\lambda}(\boldsymbol{x} \cdot \boldsymbol{y})$
  - Positive definite: $\boldsymbol{x} \cdot \boldsymbol{x} = \boldsymbol{x}^\top \boldsymbol{x} > \boldsymbol{0}, \quad \forall \boldsymbol{x} \neq \boldsymbol{0}$

$$\boldsymbol{x} \cdot \boldsymbol{x} = \boldsymbol{0} \Leftrightarrow \boldsymbol{x} = \boldsymbol{0}$$

Vector Operations

- Outer product

$$\boldsymbol{x} = [x_1 \quad x_2 \quad \cdots \quad x_m]^\top \in \mathbb{R}^m$$

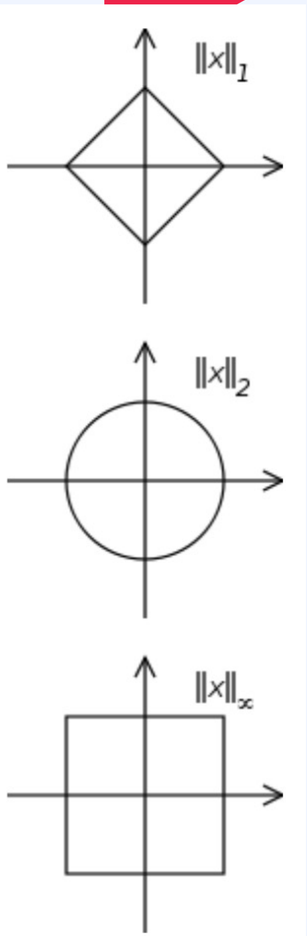
$$\boldsymbol{y} = [y_1 \quad y_2 \quad \cdots \quad y_n]^\top \in \mathbb{R}^n$$

$$\boldsymbol{x} \otimes \boldsymbol{y} = \boldsymbol{x}\boldsymbol{y}^\top = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} [y_1 \quad y_2 \quad \cdots \quad y_n] = \begin{bmatrix} x_1y_1 & x_1y_2 & \cdots & x_1y_n \\ x_2y_1 & x_2y_2 & \cdots & x_2y_n \\ \vdots & \vdots & & \vdots \\ x_ny_1 & x_ny_2 & \cdots & x_ny_n \end{bmatrix} \in \mathbb{R}^{m \times n}$$

Vector Norm

- Length of vector
- Examples
  - 1-norm:
    $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$
  - 2-norm:
    $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}$
  - ∞-norm:
    $\|\boldsymbol{x}\|_\infty = \max(|x_1|, |x_2|, \dots, |x_n|)$

p-norm
$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}$

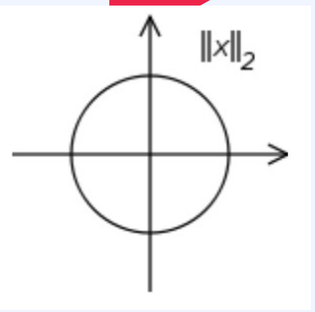


Vector Norm

- 2-norm revisited

$$\boldsymbol{x} = [x_1 \quad x_2 \quad \cdots \quad x_n]^\top \in \mathbb{R}^n$$

$$\boldsymbol{y} = [y_1 \quad y_2 \quad \cdots \quad y_n]^\top \in \mathbb{R}^n$$

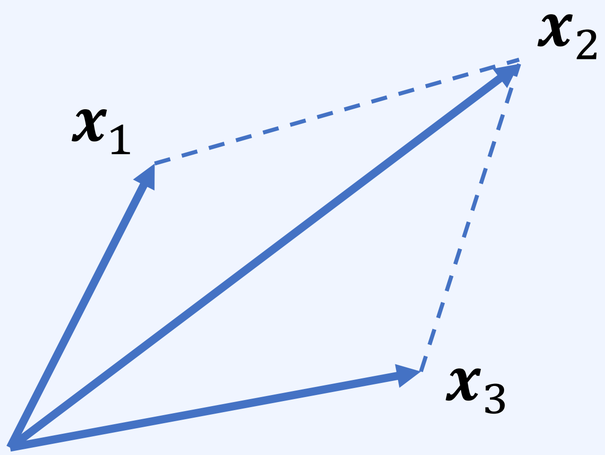


$$\|\boldsymbol{x}\|_2^2 = \sum_{i=1}^{n} |x_i|^2 = x_1^2 + x_2^2 + \cdots + x_n^2 = \boldsymbol{x}^\top \boldsymbol{x} = \boldsymbol{x} \cdot \boldsymbol{x}$$

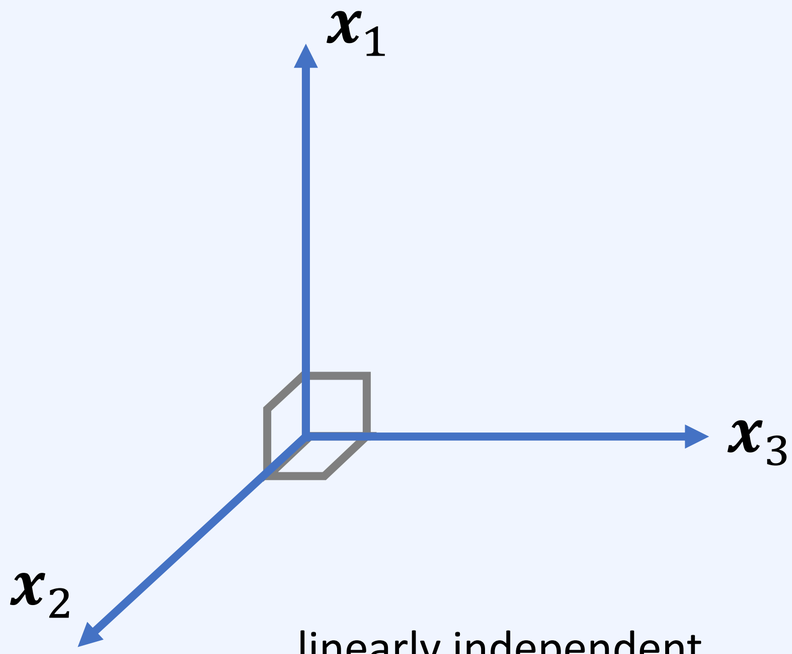
$$\|\boldsymbol{x} - \boldsymbol{y}\|_2^2 = \sum_{i=1}^{n} |x_i - y_i|^2 = (\boldsymbol{x} - \boldsymbol{y})^\top (\boldsymbol{x} - \boldsymbol{y}) = \|\boldsymbol{x}\|_2^2 - 2\boldsymbol{x}^\top \boldsymbol{y} + \|\boldsymbol{y}\|_2^2$$

Linear Dependency

- Given a set of vectors $\mathbf{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\}$
- Linear combination of vectors: $a_1\boldsymbol{x}_1 + a_2\boldsymbol{x}_2 + \cdots + a_k\boldsymbol{x}_k$
- $\boldsymbol{x}_i \in \boldsymbol{X}$ is linearly dependent if it can be represented by a linear combination of $\mathbf{X}\backslash\{\boldsymbol{x}_i\}$.



linearly dependent



linearly independent

Basis

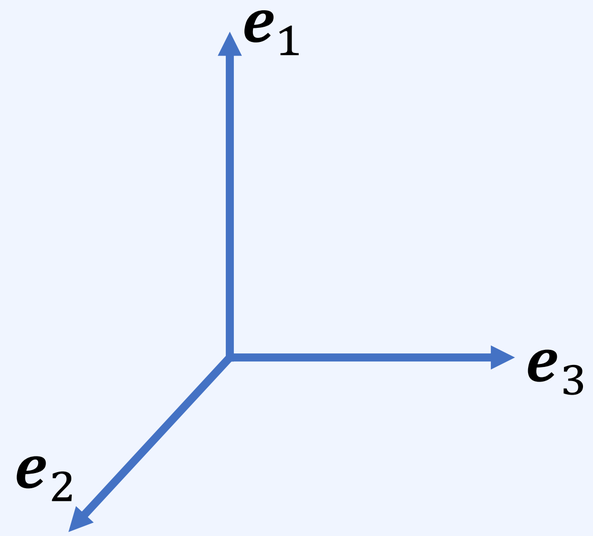
- A linearly independent set of vectors that spans the whole space.
- Standard basis (*a.k.a.* unit vectors)

$$\boldsymbol{e}_1 = \begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}, \boldsymbol{e}_2 = \begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix}, \dots, \boldsymbol{e}_n = \begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix}$$

  - Example: $[5 \quad 2 \quad 3]^\top = 5\boldsymbol{e}_1 + 2\boldsymbol{e}_2 + 3\boldsymbol{e}_3$
  - Projection of a vector:

    $\boldsymbol{x} \cdot \boldsymbol{e}_i = \boldsymbol{x}^\top \boldsymbol{e}_i$

    $\boldsymbol{x} = [\boldsymbol{x}^\top \boldsymbol{e}_1 \quad \boldsymbol{x}^\top \boldsymbol{e}_2 \quad \dots \quad \boldsymbol{x}^\top \boldsymbol{e}_n]$



- Orthogonal: $\boldsymbol{e}_i{}^\top \boldsymbol{e}_j = 0, \quad \forall i \neq j$
- Normalized: $\boldsymbol{e}_i{}^\top \boldsymbol{e}_i = 1$

} **Orthonormal**

Matrix

- Rectangular array of numbers
- Notation (표기법)

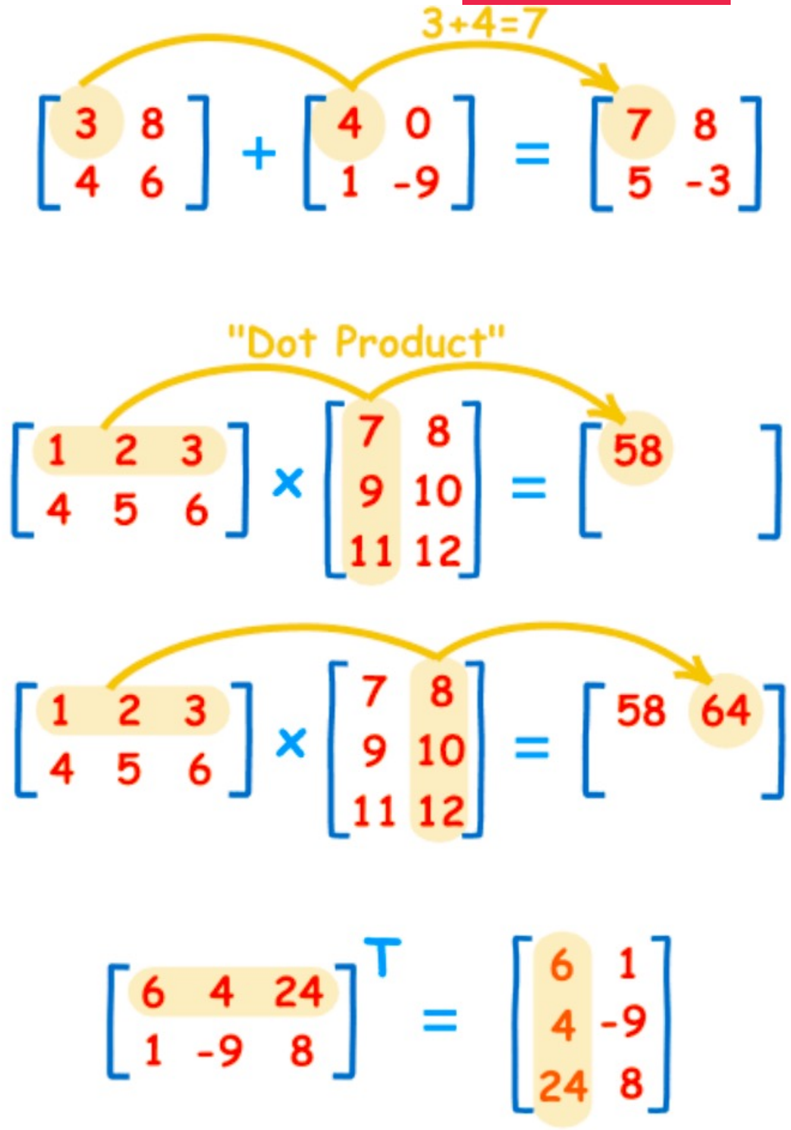
$$\boldsymbol{X} \in \mathbb{R}^{m \times n}. \quad \boldsymbol{X} = \begin{bmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_n \\ x_2 y_1 & x_2 y_2 & \cdots & x_2 y_n \\ \vdots & \vdots & & \vdots \\ x_n y_1 & x_n y_2 & \cdots & x_n y_n \end{bmatrix}. \quad [\boldsymbol{X}]_{ij} = x_{ij}.$$

- Special matrices
  - Square matrix: matrix with (# of rows) == (# of columns)
  - Identity matrix: matrix with diagonal elements ones and non-diagonal elements zeros,

$$\boldsymbol{XI} = \boldsymbol{IX} = \boldsymbol{X}$$

Matrix Operations (행렬 연산과, 그 연산들의 성질)

- Addition (덧셈)
  - Commutative(교환법칙): $\boldsymbol{X}+\boldsymbol{Y}=\boldsymbol{Y}+\boldsymbol{X}$
  - Associative(결합법칙): $(\boldsymbol{X}+\boldsymbol{Y})+\boldsymbol{Z}=\boldsymbol{X}+(\boldsymbol{Y}+\boldsymbol{Z})$
- Subtraction (뺄셈)
  - Not-commutative, not-associative
- Multiplication (곱셈)
  - Associative: $(\boldsymbol{XY})\boldsymbol{Z}=\boldsymbol{X}(\boldsymbol{YZ})$
  - Distributive: $\boldsymbol{X}(\boldsymbol{Y}+\boldsymbol{Z})=\boldsymbol{XY}+\boldsymbol{XZ}$
  - Non-commutative(결합법칙 성립안함): $\boldsymbol{XY}\neq\boldsymbol{YX}$
- Transpose: $[\boldsymbol{X}^{\mathsf{T}}]_{ij}=[\boldsymbol{X}]_{ji}$
  - $(\boldsymbol{X}^{\mathsf{T}})^{\mathsf{T}}=\boldsymbol{X}$
  - $(\boldsymbol{XY})^{\mathsf{T}}=\boldsymbol{Y}^{\mathsf{T}}\,\boldsymbol{X}^{\mathsf{T}}$

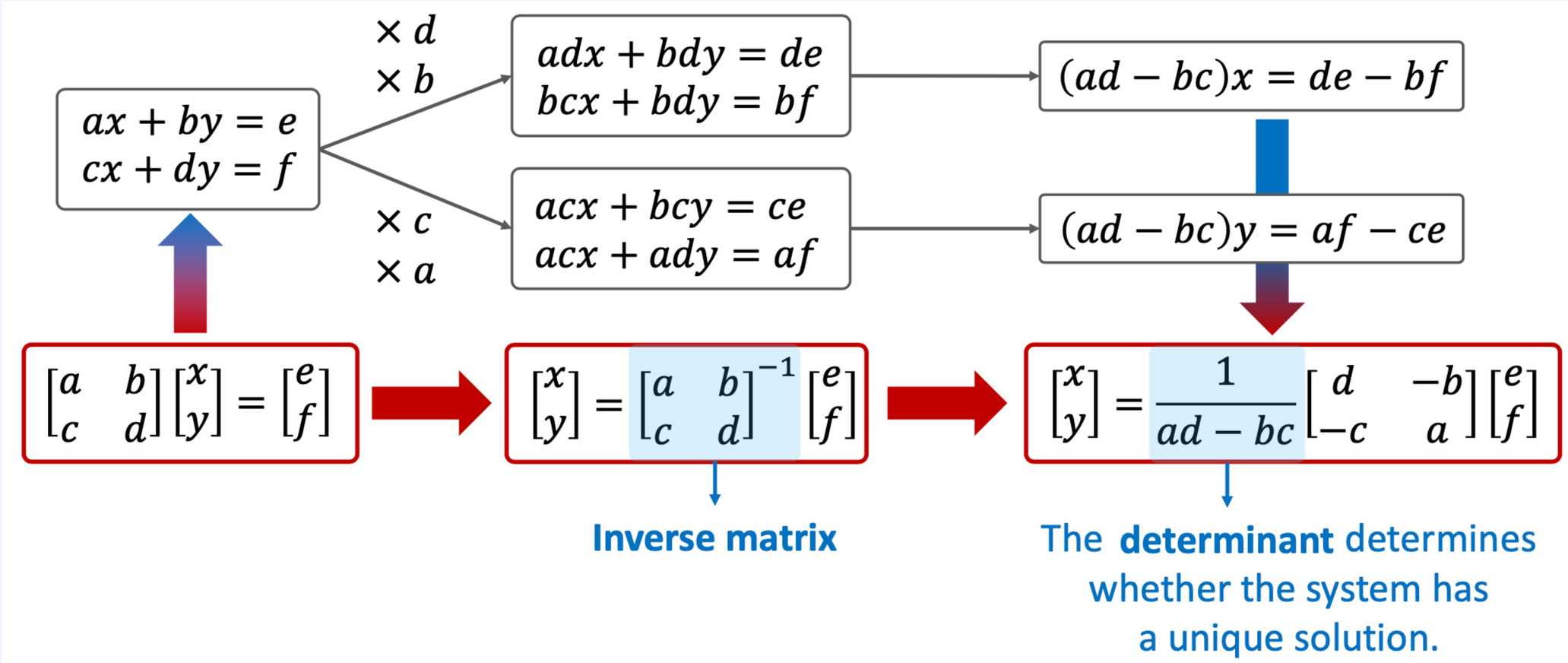


Rank of Matrix

- Definition
  - The number of linearly independent rows or columns in a matrix
- Examples
  - $\text{rank}\left(\begin{bmatrix} -1 & 2 \\ 4 & -8 \end{bmatrix}\right) = 1.$ $\text{rank}\left(\begin{bmatrix} -5 & -3 \\ 7 & 2 \end{bmatrix}\right) = 2$
- Properties
  - $\boldsymbol{X} \in \mathbb{R}^{m \times n},\ \text{rank}(\boldsymbol{X}) \leq \min(m, n)$
  - $\boldsymbol{X}$ is full-rank if $\text{rank}(\boldsymbol{X}) = \min(m, n)$
  - $\text{rank}(\boldsymbol{X}) = \text{rank}(\boldsymbol{X}^\top)$
  - $\text{rank}(\boldsymbol{XY}) \leq \min(\text{rank}(\boldsymbol{X}), \text{rank}(\boldsymbol{Y}))$
  - $\text{rank}(\boldsymbol{X} + \boldsymbol{Y}) \leq \text{rank}(\boldsymbol{X}) + \text{rank}(\boldsymbol{Y})$
  - A square matrix with full rank is non-singular;
    - otherwise, the matrix is singular.

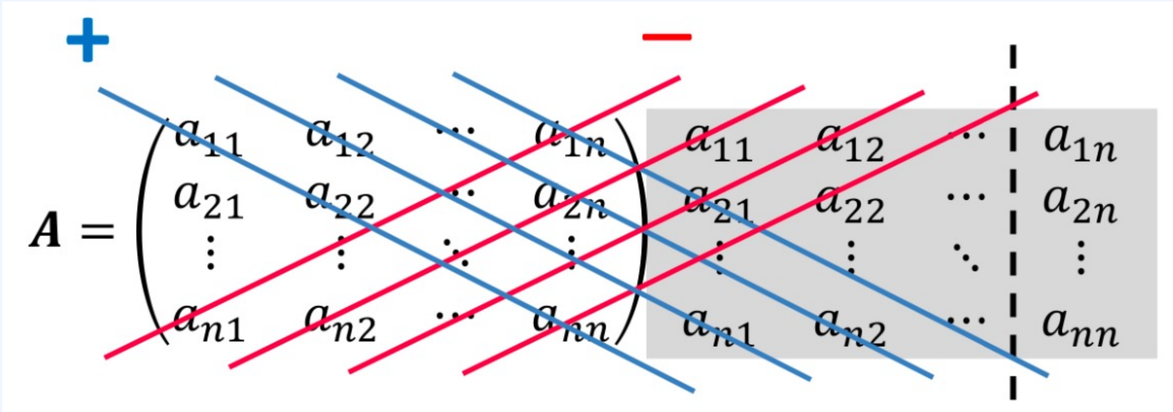
Square Matrix and Linear Equations

- Solving linear equations given by a square matrix



Determinant

- How to compute



MATLAB command: det(A)
Python command: numpy.linalg.det(A)

- Properties
  - Computed from the elements of a square matrix
  - $\det(\boldsymbol{A}) = 0$ iff $\boldsymbol{A}$ is singular.
  - $\det(\boldsymbol{AB}) = \det(\boldsymbol{A})\det(\boldsymbol{B})$
  - $\det(\boldsymbol{A^{-1}}) = \det(\boldsymbol{A})^{-1}$
  - $\det(\lambda \boldsymbol{A}) = \lambda^n \det(\boldsymbol{A})$

Inverse Matrix

- Condition to have inverse matrix: square and non-singular
- Inverse matrix of $\mathbf{X}$: $\boldsymbol{X^{-1}}$
  - $\boldsymbol{XX^{-1} = X^{-1}X = I}$
- 2D example
  - $\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$
- Properties
  - $\boldsymbol{(XY)^{-1} = Y^{-1}\,X^{-1}}$
  - $\boldsymbol{X^{-1} = X^{-1}}$
  - $\boldsymbol{(X^{\top})^{-1} = (X^{-1})^{\top}}$
  - For an **orthonormal matrix**, $\boldsymbol{X^{-1} = X^{\top}}$
  - For a diagonal matrix $\boldsymbol{D} = \mathrm{diag}(d_1, d_2, \ldots, d_n) \in \mathbb{R}^{m\times n}$
    - $\boldsymbol{D^{-1}} = \mathrm{diag}(d_1^{-1}, d_2^{-1}, \ldots, d_n^{-1})$

Orthonormal matrix

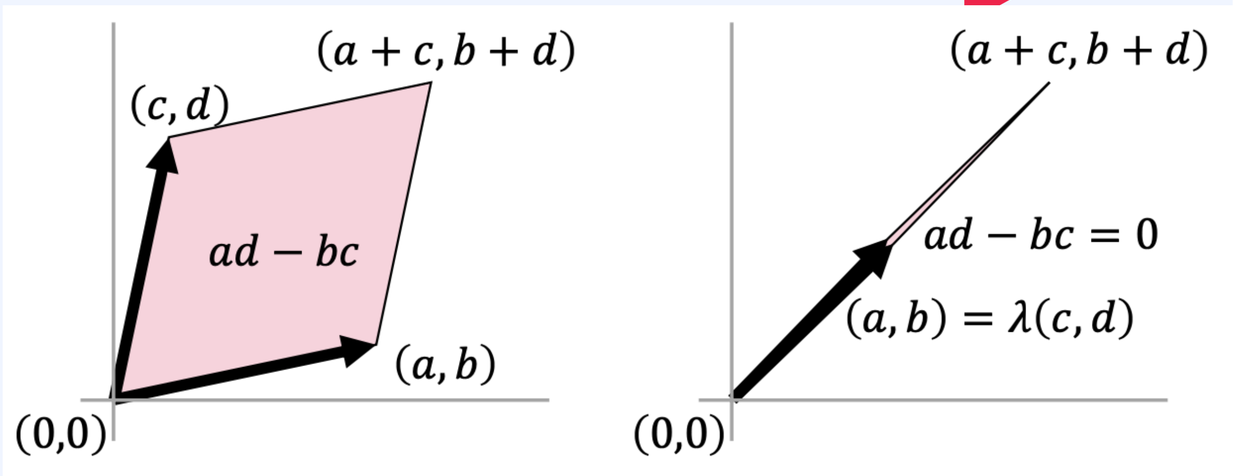
A square matrix with real entries whose columns and rows are orthonormal vectors.

- Identity matrix: $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
- Rotation matrix: $\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$

Determinant, Inverse Matrix, and Linear Equations

- $\det\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = ad - bc = 0$

- Geometric interpretation
  (singular matrix)



- No inverse matrix

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{0}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

- No solution to the linear equations

$$0 \cdot x = de - bf, \qquad 0 \cdot y = af - ce$$

Eigen Decomposition

- Eigenvector and eigenvalue
  - Defined for a square matrix $\boldsymbol{A}$
  - Non-zero vector $\boldsymbol{x}$ for which there is a $\lambda \in \mathbb{R}$ such that $\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{x}$
  - Eigenvector: $\boldsymbol{x}$
  - Eigenvalue: λ

- How to compute
  - Solve $(\boldsymbol{A} - \lambda\mathbf{I})\boldsymbol{x} = \mathbf{0}$ for $\boldsymbol{x}$ and λ

- Simply,
  - MATLAB command: $[\boldsymbol{V}, D] = \mathrm{eig}(A)$
  - Python command: $[\boldsymbol{V}, D] = \mathrm{np.linalg.eig}(A)$

Eigen Decomposition

- A square and symmetric matrix $\boldsymbol{A}$ can be decomposed as

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^{\top} = [v_1 \quad v_2 \quad \cdots \quad v_n] \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

$$v_i{}^{\top} v_i = 1$$
$$v_i{}^{\top} v_j = 0$$

  - $\boldsymbol{V}$ is an orthonormal matrix of $\boldsymbol{A}$'s eigenvectors.
  - $\boldsymbol{D}$ is an diagonal matrix of the associated eigenvalues.
- Properties
  - A square matrix is singular if any of its eigenvalues is zero.
  - A square matrix $\boldsymbol{A}$ is positive definite (negative definite) if all eigenvalues are positives (negatives).
  - Semi-definiteness: $\forall \boldsymbol{x} \neq 0, \quad \boldsymbol{x}^{\top}\boldsymbol{A}\boldsymbol{x} > \boldsymbol{0}\ (\boldsymbol{x}^{\top}\boldsymbol{A}\boldsymbol{x} < \boldsymbol{0})$
    $\forall \boldsymbol{x} \neq 0, \quad \boldsymbol{x}^{\top}\boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{0}\ (\boldsymbol{x}^{\top}\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{0})$

Eigen Decomposition

- Computing inverse matrix
  - Problem: $\boldsymbol{AX} = \boldsymbol{I}$
  - Decomposition: $\boldsymbol{A} = \boldsymbol{VDV}^{\top}$
  - Problem revisited: $\boldsymbol{A} = \boldsymbol{VDV}^{\top}\boldsymbol{X} = \boldsymbol{I}$
  - Solution: $\boldsymbol{X} = \boldsymbol{VD}^{-1}\boldsymbol{V}^{\top}$

- Solving optimization problems
  - Objective function: $\min\|\boldsymbol{Ax}\|^2$
  - Find the eigenvector of $\boldsymbol{A}^{\top}\boldsymbol{A}$ corresponding to the least eigenvalue (e.g., PCA)

$$\min\|\boldsymbol{Ax}\|^2 = \min\|\lambda\boldsymbol{x}\|^2$$

The least eigenvalue

Corresponding eigenvector

If you need more about linear algebra,

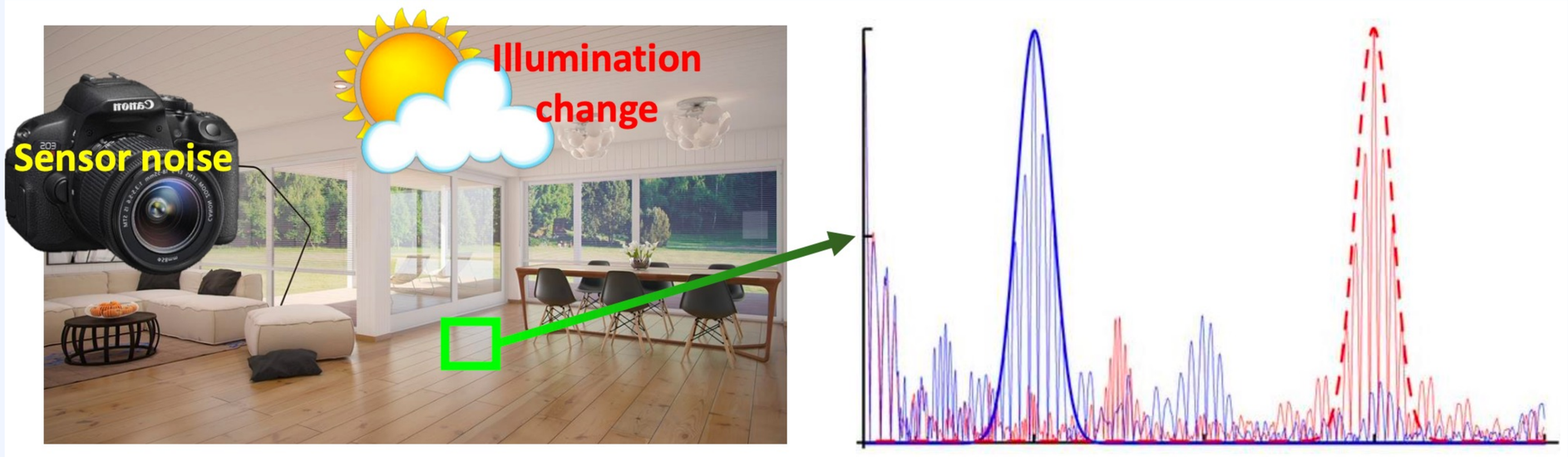
- Matrix Cookbook
  - https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf

컴퓨터 비전 초격차: 이론/실습

3 Preliminary – Probability

Probability in Computer Vision

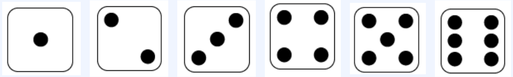
- Visual sensing is noisy and uncertain.



- Probabilistic models are useful to handle such uncertainty.

Most of the slides are inspired by POSTECH AIGS549 class (Prof. Suha Kwak)

Random variable

- Sample Space (Ω)
  - The set of all possible outcomes to an experiment
  - Example: Tossing a single six-sided dice: {  }

- Random Variable ($X: \Omega \rightarrow \mathbb{R}$)
  - A function that assigns a (usually *real*) number to each point in a sample space Ω
  - A variable whose possible values are numerical outcomes of a random phenomenon
  - Discrete or continuous
    - Examples:

$X($  $) = 3$

$X($  $) = 5$

Discrete Random Variable

The probability that X takes value ≤ 3

$= P(\omega \in \Omega : X(\omega) \leq 3)$

$= P(X \leq 3)$

Probability Axioms

- Sample space (Ω): The set of all possible outcomes

- Event space (E): A power set whose element is a set of Ω
  - If $A \in E$, then $A \subseteq \Omega$

- Probability measure (P)
  - A function $P: E \to \mathbb{R}$ with the following conditions:
    - $P(A) \geq 0, \forall A \in E$
    - $P(\Omega) = 1$
    - $\sum_i P(A_i) = P(A_1 \cup A_2 \cup A_3 \cup \cdots)$, if A_i's are disjoint

Important Properties in Probability

- Set inclusion
  - $A \subseteq B \Longrightarrow P(A) \leq P(B)$
- Union
  - $P(A \cup B) = P(A) + P(B) - P(A \cap B)$
- Complement
  - $P(\Omega \backslash A) = 1 - P(A)$

- Independence
  - If A and B are independent, $P(A \cap B) = P(A)P(B)$
- Notation
  - $P(A \cap B) = P(A, B)$

Conditional Probability

- Conditional probability of A given B
  - $P(A|B) = \frac{P(A,B)}{P(B)}$
  - $P(A,B) = P(A|B)\,P(B)$

- Independence

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(A)P(B)}{P(B)} = P(A)$$

- Conditional independence
  - A and B are conditionally independent give C.

$$P(A,B|C) = P(A|C)P(B|C)$$
$$P(A|B,C) = P(A|C)$$

$$P(\text{rain},\text{sunny})$$
$$= P(\text{sunny})P(\text{rain}|\text{sunny})$$
$$= 0.5 \times 0.2 = 0.1$$

$$P(\text{rain},\text{cloudy})$$
$$= P(\text{cloudy})P(\text{rain}|\text{cloudy})$$
$$= 0.5 \times 0.7 = 0.35$$

$$P(T,B)$$
$$= P(T)P(B|T)$$
$$= P(T)P(B)$$
$$= 0.5 \times 0.5$$

Bayes' Theorem

- Revisit conditional probabilities

$$P(A|B) = \frac{P(A,B)}{P(B)}, P(B|A) = \frac{P(A,B)}{P(A)}$$

$$P(A,B) = P(A|B)\,P(B) = P(B|A)\,P(A)$$

- Bayes' Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

posterior $\propto$ *linklihood* $\times$ *prior*

Bayes' Theorem

- Example

$$posterior \propto linklihood \times prior$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

x5 x3 x2

$[prior]$

$P(\text{apple}) = 0.5, \quad P(\text{pear}) = 0.3, \quad P(\text{orange}) = 0.2$

$[likelihood]$

$p(\text{tongue} \mid \text{apple}) = 0.3, \quad p(\text{tongue} \mid \text{pear}) = 0.5, \quad p(\text{tongue} \mid \text{orange}) = 0.9$

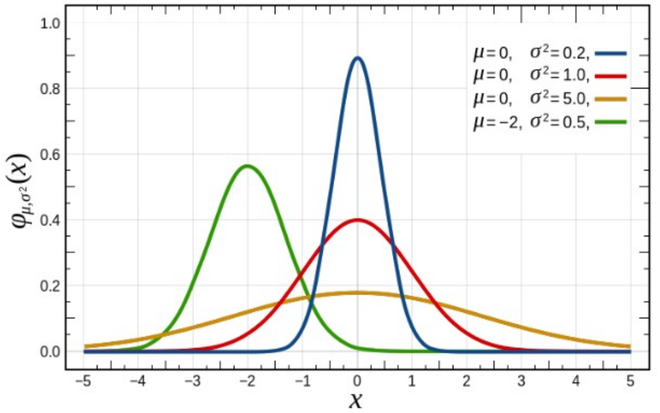
$[posterior \propto likelihood \times prior]$

$P(\text{apple} \mid \text{tongue}) \propto 0.15, \quad P(\text{pear} \mid \text{tongue}) \propto 0.15, \quad P(\text{orange} \mid \text{tongue}) \propto 0.18$

Gaussian Distributions

- Univariate Gaussian distribution

$$f(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

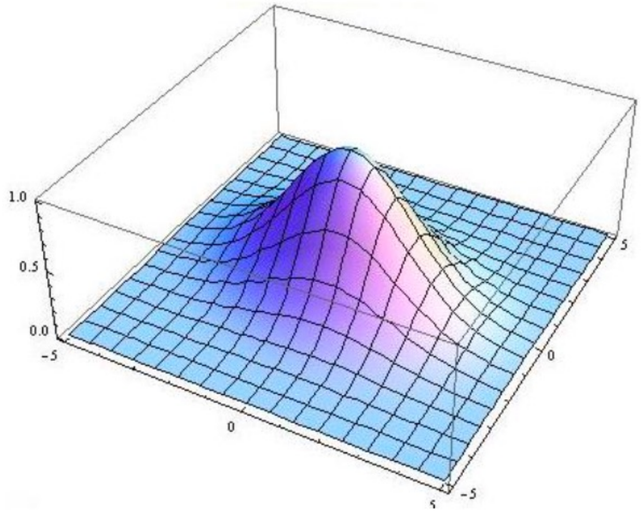


- Multivariate Gaussian distribution

$$f(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right),$$

where $\boldsymbol{x} = [x_1, x_2, \ldots, x_d]^{\top}$, $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_d]^{\top}$ and

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathrm{cov}(x_1,x_1) & \mathrm{cov}(x_1,x_2) & \cdots & \mathrm{cov}(x_1,x_d) \\ \mathrm{cov}(x_2,x_1) & \mathrm{cov}(x_2,x_2) & \cdots & \mathrm{cov}(x_2,x_d) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{cov}(x_d,x_1) & \mathrm{cov}(x_d,x_2) & \cdots & \mathrm{cov}(x_d,x_d) \end{bmatrix}.$$



Why Gaussian?

- Only two parameters to estimate
  - Mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$

- Self-conjugate
  - Gaussian x Gaussian = Gaussian
  - Bayes' theorem

컴퓨터 비전 초격차: 이론/실습

4 Preliminary – OpenCV, PIL, matplotlib

Index

- OpenCV
- matplotlib
- PIL
- Scikit-image
- Example codes

Most of the slides are inspired by POSTECH AIGS549 class (Prof. Suha Kwak)

OpenCV

- 컴퓨터 비전 을 목적으로 하는 오픈 소스 라이브러리
- 인텔 CPU 에서 사용하는 경우 속도 향상을 볼 수 있는 IPP (Intel Performance Primitives) 를 지원한다.
- 기존에 C++ 에서 사용할 수 있게 구현되었으나, OpenCV-python 을 통해 python 포팅도 되어 있다.
- 단점: **GPU operation** 에 대한 지원을, 명시적으로 python 과 연계하여 하지는 않는다.
- 설치: pip install opencv-python
- OpenCV documentation: https://docs.opencv.org/4.x/
- OpenCV python tutorial: https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html

Matplotlib

- Python 과 numpy array를 기반으로 plotting 과 visualize를 목적으로 하는 라이브러리
- 주어진 데이터에 대해서 차트와 plot 을 편리하게 그려주는 데이터 시각화 패키지 이다.
- 우리는 본 강의에서 이미지 시각화와 디버깅을 위해 사용한다.

- 설치 : pip install matplotlib

- Matplotlib documentation : https://matplotlib.org/stable/index.html
- Mattploblit tutorials: https://matplotlib.org/stable/tutorials/index.html

https://ko.wikipedia.org/wiki/Matplotlib

OpenCV

- Example 1 : Image read and visualize
- Example 2 : multiple image visualize, figure/axs in matplotlib
- Example 3 : Image cropping, masking and save
- Example 4 : Image matching and visualize

PIL

- PIL (Python Imaging Library) 로서 컴퓨터 비전 보다는 이미지 처리에 중점을 둔 라이브러리.

- 픽셀단위의 이미지 조작이나, 마스킹, 투명도 제어, 윤곽 보정 및 검출 등의 다양한 이미지 조작을 할 수 있다.

- 특히,

- 설치 : pip install pillow
- Pillow (PIL forked) documentation: https://pillow.readthedocs.io/en/stable/

PIL

- Image read and visualize
- Image cropping, rotating, scaling
- Image interpolation (upsampling, downsampling)

Scikit-image

- Scikit-image 는 Pillow (PIL) 과 마찬가지로, 이미지 조작과 필터링이 가능하다.

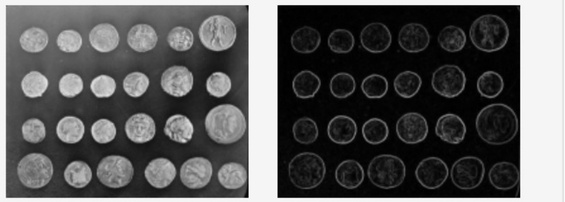
- numpy 를 기반으로 동작하기 때문에, 좀 더 numpy와의 호환성이 좋다.

Getting Started

Filtering an image with `scikit-image` is easy! For more examples, please visit our gallery.

```
from skimage import data, io, filters

image = data.coins()
# ... or any other NumPy array!
edges = filters.sobel(image)
io.imshow(edges)
io.show()
```



- 설치 : pip install scikit-image
- Documentation: https://scikit-image.org/

Environment

- Jupyter notebook

- If you don’t have GPU resources, please use Google CoLab!!

컴퓨터 비전 초격차: 이론/실습

5 Preliminary – PyTorch, Torchvision, conda

Index

- PyTorch
- TorchVision
- Anaconda
- 실습
  - Installation
  - Tensors
  - Dataloader
  - Network
  - Training
  - Save and Load Model Weights
  - Transfer learning

Most of the slides are inspired by POSTECH
AIGS549 class (Prof. Suha Kwak)

PyTorch

- GPU와 CPU를 사용해서 deep learning 을 하기 위해 최적화된 tensor library이다.
- 원래 Lua라는 프로그래밍 언어로 된 Torch였으나, 오픈소스로 여러사람이 개발하다가 facebook AI 가 개발에 참여하면서, 사용성이 좋아지고 있다.
- PyTorch로 넘어오면서 특히 좋은 점은 자동 미분 모듈이 있어서 따로 backward함수를 구현하지 않아도 된다는 점이다.
- TensorFlow 와 비슷하게 많이 사용되나, 본 강의에서는 PyTorch위주로 다룬다.
- 모듈, tensorflow와의 비교, 장점: https://ko.wikipedia.org/wiki/PyTorch
- 설치: https://pytorch.org/get-started/locally/ : 각자 머신 스펙에 맞게 설치.
- PyTorch documentation: https://pytorch.org/docs/stable/index.html
- PyTorch tutorial: https://pytorch.org/tutorials/

TorchVision

- PyTorch project 의 일부로서 이미지를 처리하는 다양한 인터페이스(함수, 클래스 …)를 제공한다.

- Numpy array (또는 PIL) 이미지의 torch.Tensor type으로의 형변환해주는 함수 (transforms.ToPILImage(), transforms.ToTensor())와,
- image를 자르거나 photometric jitering을 주는 등의data augmentation 기능을 제공한다.

- 또한, 대표적인 CNN model과 그 pre-trained weights을 쉽게 사용하도록 해준다.

- 설치: 보통 pytorch 설치할 때에 같이 설치한다.

- TorchVision documentation: https://pytorch.org/vision/stable/index.html

- 추가적으로 image augmentation 할 때에 사용하는 albumentations 라이브러리가 있다.
- 예시: https://albumentations.ai/. Documentation: https://albumentations.ai/docs/

Anaconda (시작하기에 앞서)

- 아나콘다 : 패키지 (라이브러리 및 dependencies) 관리를 쉽게하기 위해 사용하는 오픈소스 패키지 관리 시스템.

- 도커나 다른 방식을 사용하기도 하지만, 보통 anaconda 를 사용한 패키지 관리가 간단하고 편리하기에 많이 사용한다.

- Conda 와 같은 package manager를 쓰면 프로젝트마다 패키지나 버전 관리가 간단하고, 프로그래밍 그 자체에 좀 더 집중할 수 있게 도와준다.

- 설치: https://www.anaconda.com/products/individual
- 위에서 installer 설치 후
  bash Anaconda-latest-Linux-x86_64.sh 또는 GUI 로 설치.

- 새로운 environment 만들기.
  conda create –n {desired name} python={desired version}

Anaconda (환경 activate and remove)

- Using an environment
  source activate {desired env name} source deactivate

- Remove an environment
  conda remove –n {desired env name} --all

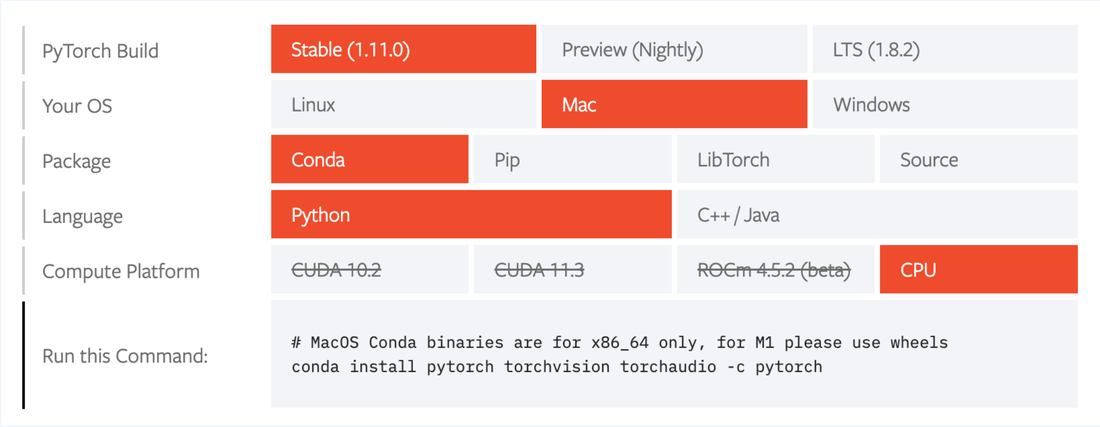
- More informations -> https://docs.conda.io/projects/conda/en/latest/user-guide/install/index.html

PyTorch and Torchvision 실습

- Installation
- Tensors
- Loading data
- Build the neural network
- Training / testing
- Model load and save
- Transfer learning (with torchvision model)

PyTorch 설치

- Installation according to your system environment



- Example: install pytorch on “Linux”, “Conda” with “CUDA 10.2”
  conda install pytorch torchvision torchaudio cudatoolkit=10.2 -c pytorch
- *When you use conda, CUDA environment also be installed (check GPU compatibility).

Tensors: creation

- What is tensors
  - Core data structure in pytorch similar to arrays (Inputs, outputs, the model parameters).
  - Very similar to numpy ndarrys (but tensors can run on GPUs)
    - *If you're familiar with ndarrays, you'll already familiar with most of the Tensor API.

- Rest of the tutorial includes:

```
import torch
import numpy as np
```

- Tensor creation
  - Directly from data

```
data = [[1, 2], [3, 4]]
x_data = torch.tensor(data) # 2x2 tensor
```

```
x_data = torch.tensor(3.14159) # scalar tensor
```

```
x_data = torch.tensor([]) # empty tensor
```

https://pytorch.org/tutorials/beginner/blitz/tensor_tutorial.html#sphx-glr-beginner-blitz-tensor-tutorial-py

Tensors: creation

- From numpy arrays

```
np_array = np.array(data)
x_np = torch.from_numpy(np_array)
```

- Tensor creation using APIs

```
>>> shape = (2, 3,)
>>> torch.rand(shape)
 tensor([[0.9864, 0.2620, 0.1767],
         [0.0322, 0.4785, 0.9585]])
>>> torch.ones(shape)
 tensor([[1., 1., 1.],
         [1., 1., 1.]])
>>> torch.zeros(shape)
 tensor([[0., 0., 0.],
         [0., 0., 0.]])
```

- Tensor creation from other tensors

```
x_ones = torch.ones_like(x_data) # retains the properties of x_data
x_ones = torch.rand_like(x_data, dtype=torch.float) # overrides the datatype of x_data
```

https://pytorch.org/tutorials/beginner/blitz/tensor_tutorial.html#sphx-glr-beginner-blitz-tensor-tutorial-py

Tensors: attributes and GPU

- Tensor attributes (type 이나 현재 처리되는 Device)

```
>>> tensor = torch.rand(3, 4)
>>> tensor.shape
torch.Size([3, 4])
>>> tensor.dtype
torch.float32
>>> tensor.device
cpu
```

- Run tensor on the GPU

```
>>> tensor = tensor.to('cuda')
>>> tensor.device
cuda:0
```

✓ operation runs on GPUs if the input tensor is on the cuda device
✓ all operand tensors to be on the same device

https://pytorch.org/docs/stable/tensors.html

Tensors: operations

- Tensor operations

```
>>> tensor = torch.ones(4, 4)
>>> tensor[:,1] = 0 # indexing
>>> tensor
tensor([[1., 0., 1., 1.],
        [1., 0., 1., 1.],
        [1., 0., 1., 1.],
        [1., 0., 1., 1.]])
>>> torch.cat([tensor, tensor, tensor], dim=1) # joining
tensor([[1., 0., 1., 1., 1., 0., 1., 1., 1., 0., 1., 1.],
        [1., 0., 1., 1., 1., 0., 1., 1., 1., 0., 1., 1.],
        [1., 0., 1., 1., 1., 0., 1., 1., 1., 0., 1., 1.],
        [1., 0., 1., 1., 1., 0., 1., 1., 1., 0., 1., 1.]])
>>> tensor + tensor * tensor - tensor # element-wise arithmetic operations
tensor([[1., 0., 1., 1.],
        [1., 0., 1., 1.],
        [1., 0., 1., 1.],
        [1., 0., 1., 1.]])
>>> tensor @ tensor # matrix multiplication
 tensor([[3., 3., 3., 3.],
        [3., 3., 3., 3.],
        [3., 3., 3., 3.],
        [3., 3., 3., 3.]])
```


https://pytorch.org/docs/stable/tensors.html

Loading data

Datasets and DataLoader

- All datasets are subclass of torch.utils.data.Dataset
  - ✓ You can make your own datasets by inheriting Dataset class
  - ✓ Some famous datasets is provided in torchvision.datasets
- DataLoader reads datasets and handles multi-process mini-batch data loading

```
# Training cycle
for epoch in range(training_epochs):
    # Loop over all batches
    for i in range(total_batch):
```

288

In the neural network terminology:

- one **epoch** = one forward pass and one backward pass of *all* the training examples
- **batch size** = the number of training examples in one forward/backward pass. The higher the batch size, the more memory space you'll need.
- number of **iterations** = number of passes, each pass using [batch size] number of examples. To be clear, one pass = one forward pass + one backward pass (we do not count the forward pass and backward pass as two different passes).

Example: if you have 1000 training examples, and your batch size is 500, then it will take 2 iterations to complete 1 epoch.

https://pytorch.org/tutorials/beginner/basics/data_tutorial.html

Loading data

Datasets from torchvision

- Example FashionMNIST

```
from torch.utils.data import Dataset
from torchvision import datasets
from torchvision.transforms import ToTensor
import matplotlib.pyplot as plt

training_data = datasets.FashionMNIST(
    root="data",
    train=True,
    download=True,
    transform=ToTensor())

test_data = datasets.FashionMNIST(
    root="data",
    train=False,
    download=True,
    transform=ToTensor())
```

- Caltech
- CelebA
- CIFAR
- Cityscapes
- COCO
- EMNIST
- FakeData
- Fashion-MNIST
- Flickr
- HMDB51
- ImageNet
- Kinetics-400
- KITTI
- KMNIST
- LSUN
- MNIST
- Omniglot
- PhotoTour
- Places365
- QMNIST
- SBD
- SBU
- SEMEION
- STL10
- SVHN
- UCF101
- USPS
- VOC
- WIDERFace

Find your dataset from
https://pytorch.org/vision/stable/datasets.html

- The dataset will be dowloaded by torchvision.
- More data augmentation techniques are in the reference page.

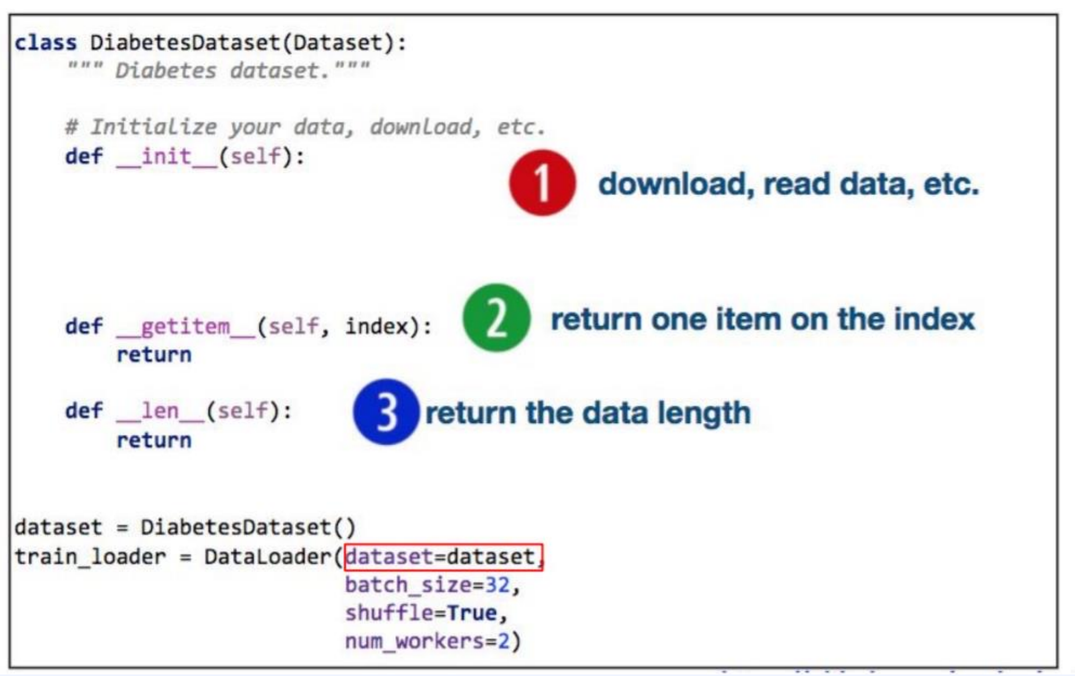
https://pytorch.org/tutorials/beginner/data_loading_tutorial.html

Loading data

Custom dataset

- Methods you should define when inherit from torch.utils.Dataset

✓ __init__(self, []) : 데이터셋 루트 디렉토리 / transform / 이미지 path ...
✓ __len__(self) : 데이터셋의 length (크기) 반환.
✓ __getitem__(self, idx) : (idx)th data 를 return. *Dataloader call this method



ner/data_loading_tutorial.html

Loading data

Dataloader

- Methods reads datasets and handles **multi-process mini-batch data loading**
  - ✓ Multi-process mini-batch loader can avoid bottleneck on reading and transferring data.
  - ✓ DataLoader is an iterable that abstracts this complexity for us in an easy API.

```
from torch.utils.data import DataLoader

train_dataloader = DataLoader(training_data, batch_size=64, shuffle=True)
test_dataloader = DataLoader(test_data, batch_size=64, shuffle=True)
```

<Needs to transfer data from storage into GPU>

Build the Neural Network

You should define two method in network class: “__init__” and “forward”

- - “__init__” : you need to implement the network architecture using torch.nn namespace .
- - “forward” : define how do you compute the forward path of the neural network.

```
class NeuralNetwork(nn.Module):
    def __init__(self):
        super(NeuralNetwork, self).__init__()
        self.flatten = nn.Flatten()
        self.linear_relu_stack = nn.Sequential(
            nn.Linear(28*28, 512),
            nn.ReLU(),
            nn.Linear(512, 512),
            nn.ReLU(),
            nn.Linear(512, 10),
        )

    def forward(self, x):
        x = self.flatten(x)
        logits = self.linear_relu_stack(x)
        return logits
```

Training

- Training data로 network network 모델 학습하기.

```
1. Define Model
2. Define Dataloader
3. Define Optimizer
4. Iterate the below lines until reaching the max_epoch.
        (1) Sample a batch from the data loader
        (2) Predict the output using the neural network
        (3) Compute the error between the answer and prediction
        (4) Compute the gradient using optimizer.
        (5) Backpropagate the gradient to optimize the NN parameters.
```

Training

1. Define Model
2. Define Dataloader
3. Define Optimizer
4. **Iterate the below lines until reaching the max_epoch.**
   - **(1) Sample a batch from the data loader**
   - **(2) Predict the output using the neural network**
   - **(3) Compute the error between the answer and prediction**
   - **(4) Compute the gradient using optimizer.**
   - **(5) Backpropagate the gradient to optimize the NN parameters.**

```python
def train_loop(dataloader, model, loss_fn, optimizer):
    size = len(dataloader.dataset)
    for batch, (X, y) in enumerate(dataloader):
        # Compute prediction and loss
        pred = model(X)
        loss = loss_fn(pred, y)

        # Backpropagation
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        if batch % 100 == 0:
            loss, current = loss.item(), batch * len(X)
            print(f"loss: {loss:>7f}  [{current:>5d}/{size:>5d}]")
```

세 줄로 이루어지는 optimization step.

https://pytorch.org/tutorials/beginner/basics/optimization_tutorial.html

Save and Load Model Weights

- Pytorch library also provide the functions(**torch.save, torch.load**) to save and store model weights.

```
model = models.vgg16(pretrained=True)
torch.save(model.state_dict(), 'model_weights.pth')
```

```
model = models.vgg16() # we do not specify pretrained=True, i.e. do not load default weights
model.load_state_dict(torch.load('model_weights.pth'))
model.eval()
```

https://pytorch.org/tutorials/beginner/basics/saveloadrun_tutorial.html

Transfer learning (with torchvision model)

- 모델을 scratch 부터 학습하는 것은 데이터도 많이 필요하고, 학습이 수렴하기까지 시간도 오래 걸린다.
- 이 때, pre-trained model을 사용하면 효율적으로 모델 학습을 할수 있다.
- 다음은 pre-trained model 을 사용한 image classification model fine-tuning (transfer learning) 예시이다.

```
model_ft = models.resnet18(pretrained=True)
num_ftrs = model_ft.fc.in_features
# Here the size of each output sample is set to 2.
# Alternatively, it can be generalized to nn.Linear(num_ftrs, len(class_names)).
model_ft.fc = nn.Linear(num_ftrs, 2)

model_ft = model_ft.to(device)

criterion = nn.CrossEntropyLoss()

# Observe that all parameters are being optimized
optimizer_ft = optim.SGD(model_ft.parameters(), lr=0.001, momentum=0.9)

# Decay LR by a factor of 0.1 every 7 epochs
exp_lr_scheduler = lr_scheduler.StepLR(optimizer_ft, step_size=7, gamma=0.1)
```

https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html.

컴퓨터 비전 초격차: 이론/실습

6 Preliminary – logging, wandb, kornia

6

Index

- logging
- wandb
- kornia

logging

- 컴퓨터비전 / 딥러닝 모델 개발 뿐 아니라 모든 프로그래밍에 있어서 logging 모듈을 잘 구현하는 것은 매우 중요하다.
- Python 에도 logging library 가 있다.
- 다양한 이벤트 수준에 대해서 정의할 수 있고, logger 와 handler가 이를 적절히 처리해 준다.
- 로깅에 대해 더 자세히 설명하는 것은 이 강의의 범위를 벗어나므로 생략한다.
- 설치: pip install logging
- Documentation: https://docs.python.org/ko/3/howto/logging.html

wandb

- Wandb (weights & Biases)는 학습 모델 개발 시에 하이퍼파라미터나, 시스템 메트릭, 결과를 로깅해주는 패키지 이다.
- 비슷하게는 tensorboard/tensorboardX나 hydra 같은 패키지들이 있다.
- Wandb 의 장점은 웹서비스 를 통해서 다양한 모델의 결과를 확인/관리 할 수 있다는 점이다.
- 또한, sweep 이라는 기능은 autoML (grid search, Bayesian optimization 등) 을 통해 하이퍼파라미터 서치를 지원해준다. 베스트 모델을 찾기 위해서 하이퍼파라미터 서치가 필수적인데, 이때 매우 유용하다.
- 설치 : pip install wandb
- Documentation: https://docs.wandb.ai/

Kornia

- Kornia 는 AI를 위한 컴퓨터비전 알고리즘의 State-of-the-art 구현체를 모아놓은 라이브러리로 비교적 최신인 2019년부터 개발되기 시작하였다.
- 특히 딥러닝 기반의 컴퓨터비전 모델을 연구/개발할 때 유용하다.
- 구현하기에 까다로운 보조함수들이 구현되어 있고, GPU 기반의 deep learning 프레임워크인 PyTorch와도 호환가능하다.
- 설치: pip install kornia
- Documentation: https://kornia.readthedocs.io/en/latest/
- Github: https://github.com/kornia/kornia