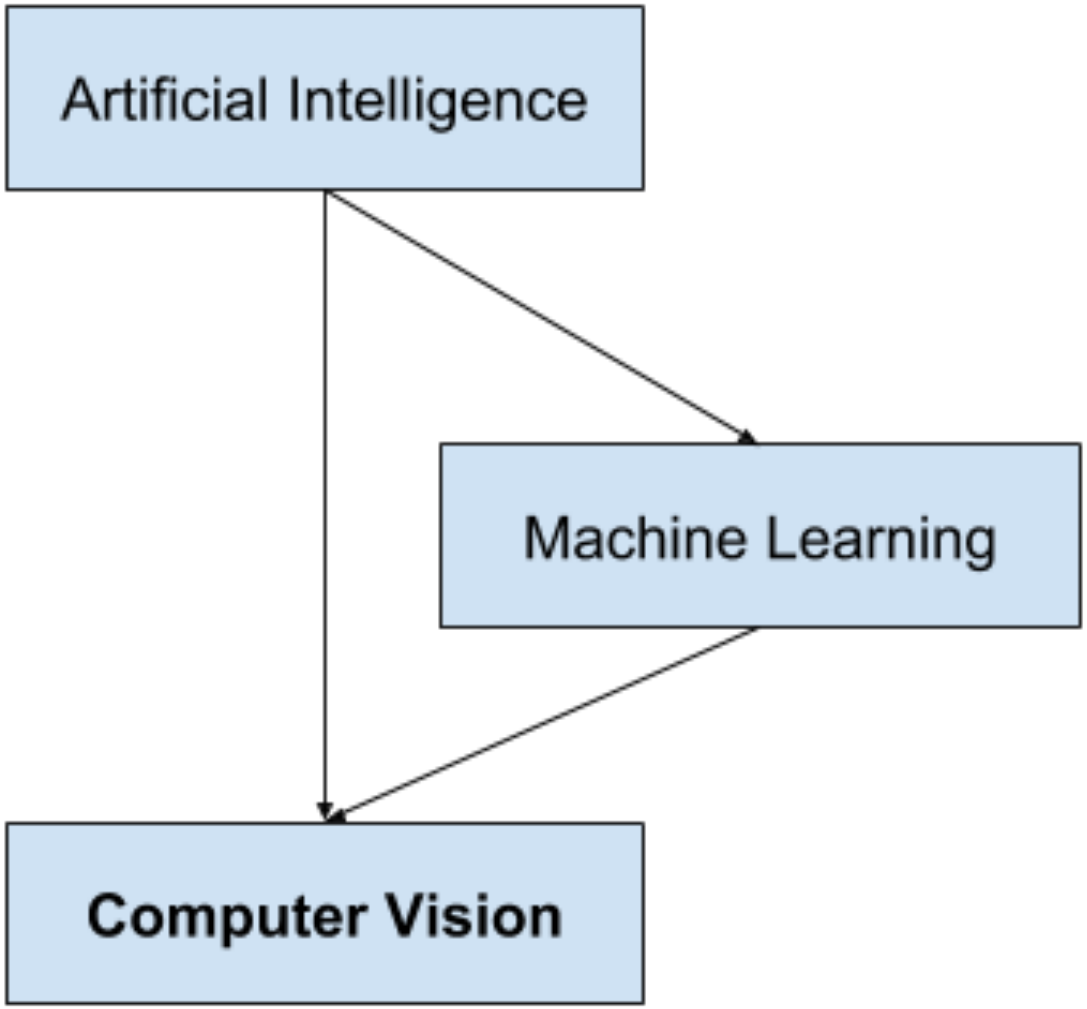
Computer Vision

Computer (Machine) vision is the automated extraction of information from images. At an abstract level, the goal of computer vision problems is to use the observed image data to infer something about the world. Information can mean anything from 3D models, camera position, object detection and recognition to grouping and searching image content.

Computer vision as a field is an intellectual frontier. Like any frontier, **it is exciting and disorganized, and there is often no reliable authority to appeal to. Many useful ideas have no theoretical grounding, and some theories are useless in practice; developed areas are widely scattered, and often one looks completely inaccessible from the other.**


Artificial Intelligence
Machine Learning
Computer Vision

Graphic vs Image

A graphic (image) is a representation of a 3D object. a digital representation of non-text information such as a drawing, chart, or photo.

An image (still Image, digital image) is the graphic converted for 2d display. the binary representation of any type of visual information like drawings, individual video frames, logos, pictures, graphs etc.

c.f) Rendering - an operation that takes graphics data and produces image data.

GRAPHIC / VECTOR

100% 400%

IMAGE / RASTER

100% 400%

Raster Graphics	Vector Graphics
They are composed of pixels.	They are composed of paths.
In Raster Graphics, refresh process is independent of the complexity of the image.	Vector displays flicker when the number of primitives in the image become too large.
Graphic primitives are specified in terms of end points and must be scan converted into corresponding pixels.	Scan conversion is not required.
Raster graphics can draw mathematical curves, polygons and boundaries of curved primitives only by pixel approximation.	Vector graphics draw continuous and smooth lines.
Raster graphics cost less.	Vector graphics cost more as compared to raster graphics.
They occupy more space which depends on image quality.	They occupy less space.
File extensions: .BMP, .TIF, .GIF, .JPG	File Extensions: .SVG, .EPS, .PDF, .AI, .DXF

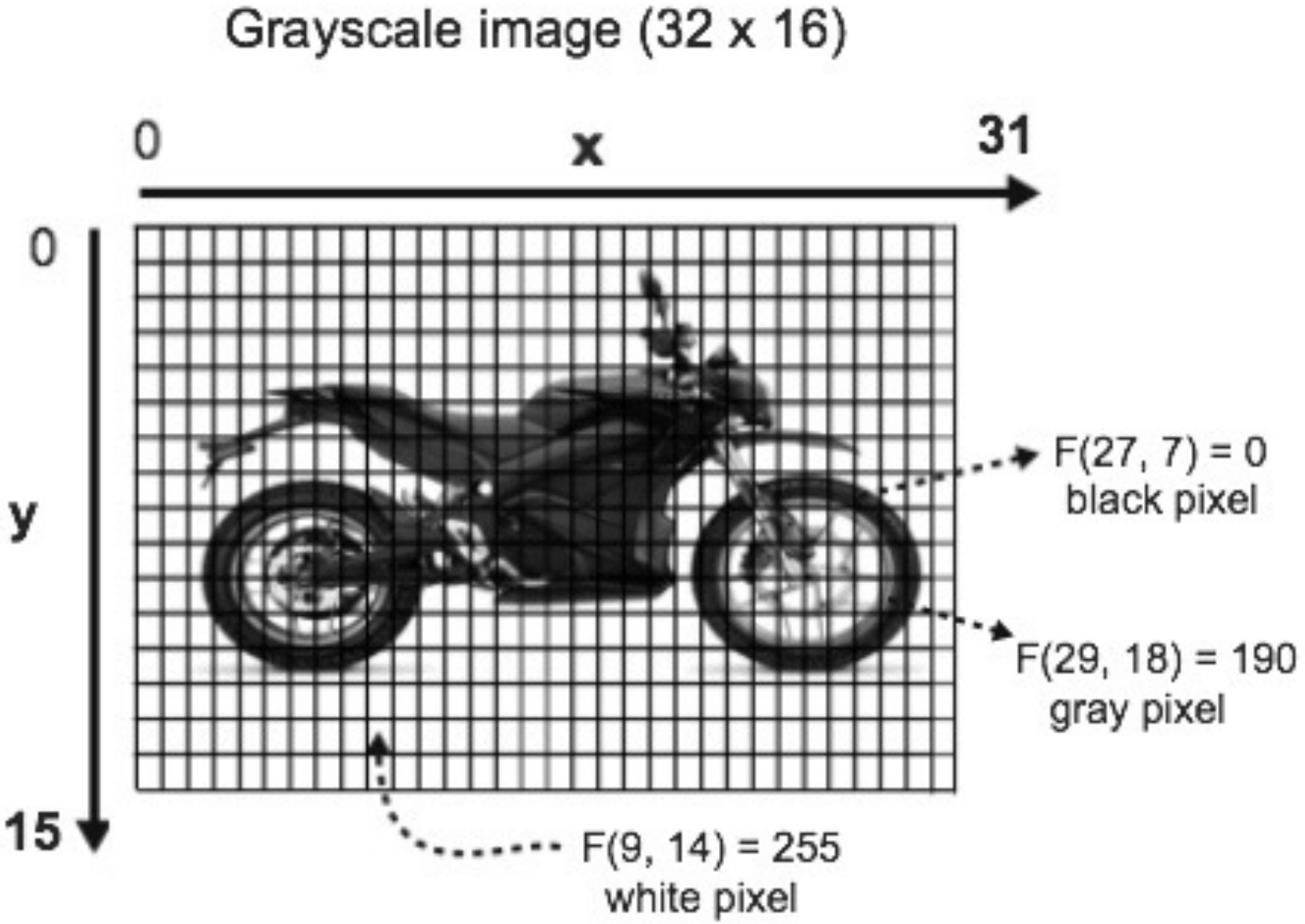
Image

An image is defined as a two-dimensional **function**, **F(x,y)**, where x and y are spatial coordinates, and the amplitude of **F** at any pair of coordinates (x,y) is called the **intensity** of that image at that point. When x,y, and amplitude values of **F** are finite, we call it a **digital image**.

Image

= function

= array


Grayscale image (32 x 16)
0
x
31
0
y
15
F(27, 7) = 0
black pixel
F(29, 18) = 190
gray pixel
F(9, 14) = 255
white pixel

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(N-1,1) & \dots & f(M-1,N-1) \end{bmatrix}$$

What we see

What computers see

Color image

$F(0, 0) = [11, 102, 35]$

RGB channels

Channel 3
Blue intensity
values

Channel 2
Green intensity
values

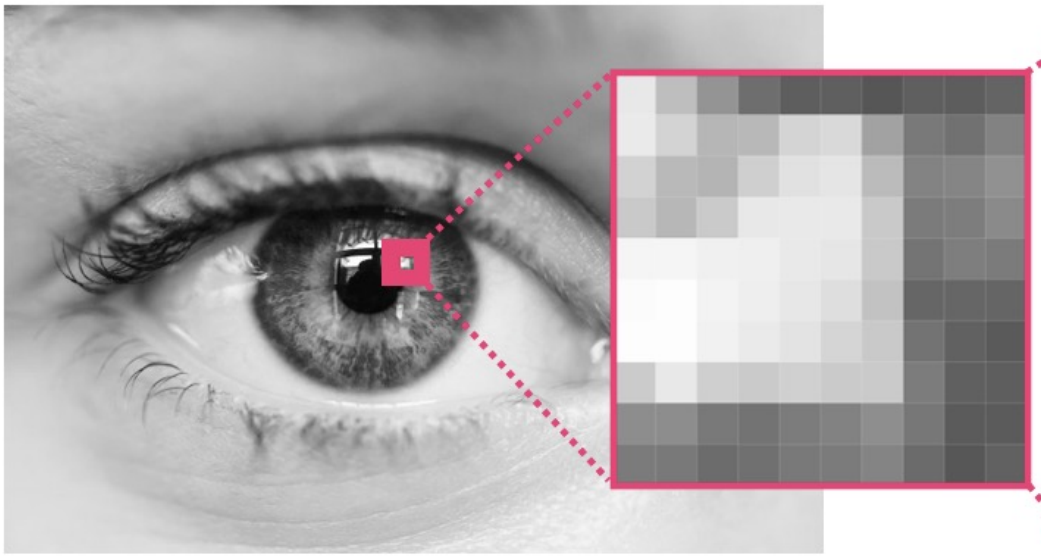
Channel 1
Red intensity
values

11	158	156	158	158	...
159	159	157	159	159	...
149	151	155	158	159	...
146	146	149	153	158	...
145	143	143	148	158	...
...	...	...	...	...	...

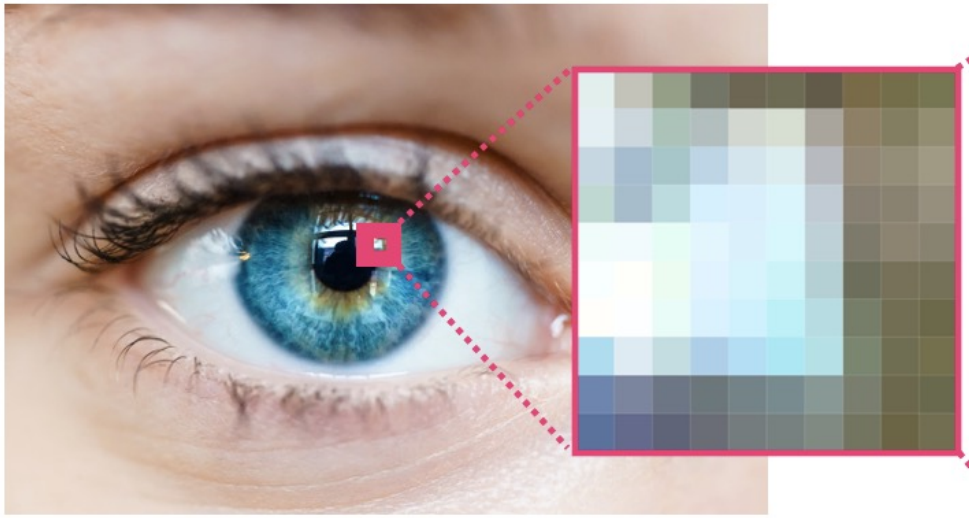
102	169	167	169	169	...
170	170	168	170	170	...
				170	...
				168	...
				168	...
				...	...

35	165	163	165	158	...
166	166	164	166	159	...
				159	...
				158	...
				167	...
				...	...

Red + Green + Blue = Forest Green (11, 102, 35)



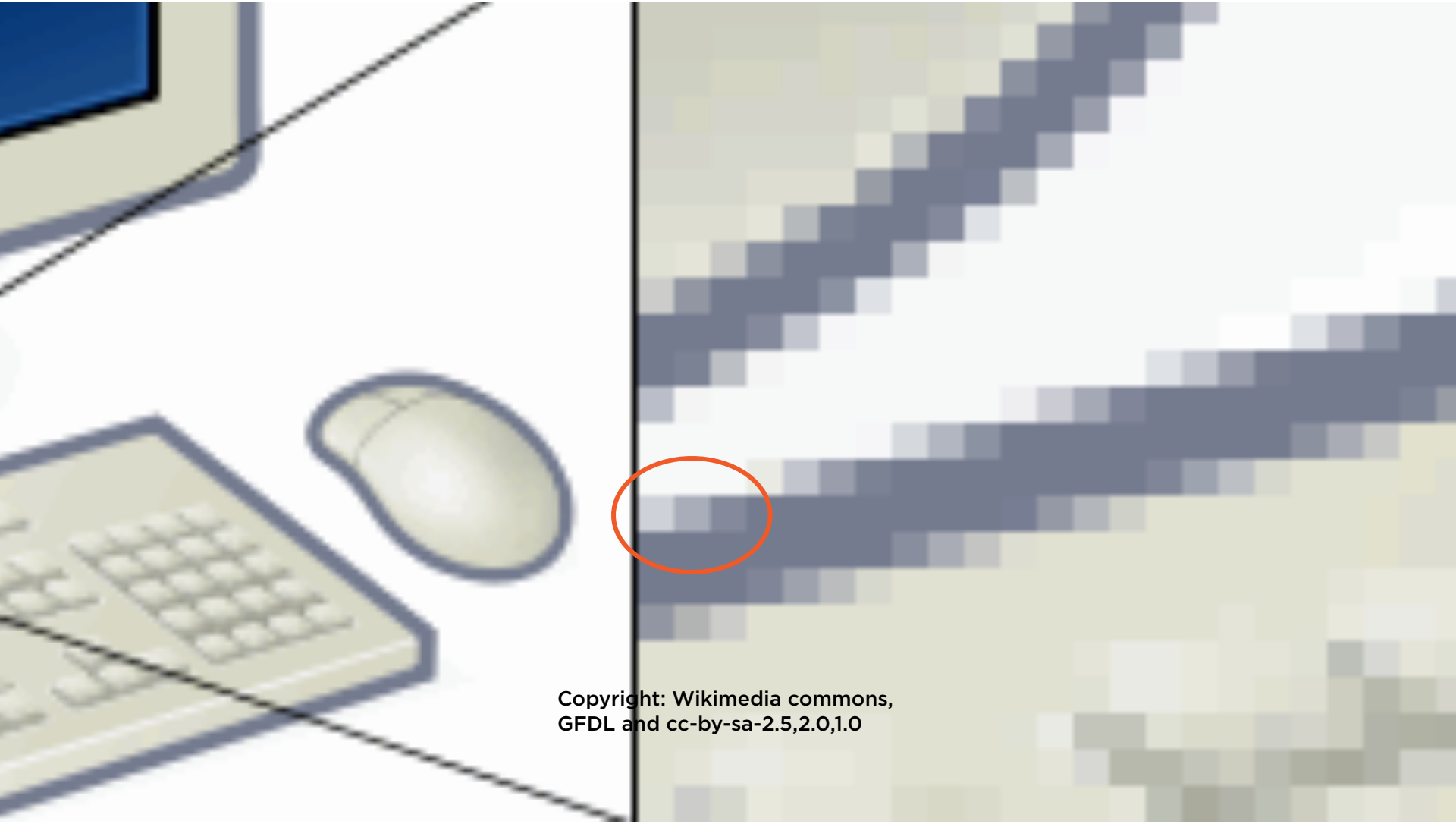
230	194	147	108	90	98	84	96	91	101
237	206	188	195	207	213	163	123	116	128
210	183	180	205	224	234	188	122	134	147
198	189	201	227	229	232	200	125	127	135
249	241	237	244	232	226	202	116	125	126
251	254	241	239	230	217	196	102	103	99
243	255	240	231	227	214	203	116	95	91
204	231	208	200	207	201	200	121	95	95
144	140	120	115	125	127	143	118	92	91
121	121	108	109	122	121	134	106	86	97



	233	188	137	96	90	95	63	73	73	82
237	202	159	120	105	110	88	107	112	121	109
226	191	147	110	101	112	98	123	110	119	142	131
221	191	176	182	203	214	169	144	133	145	155	122
185	160	161	184	205	223	186	137	147	161	140	115
181	174	189	207	206	215	194	136	142	151	133	87
246	237	237	231	208	206	192	122	143	144	111	74
254	254	241	224	199	192	181	99	122	117	107	74
239	248	232	207	187	182	184	110	114	110	113	74
193	215	193	167	158	164	181	114	112	111	105	82
113	119	110	111	113	123	135	120	108	106	113	
93	97	91	103	107	111	122	112	104	114		

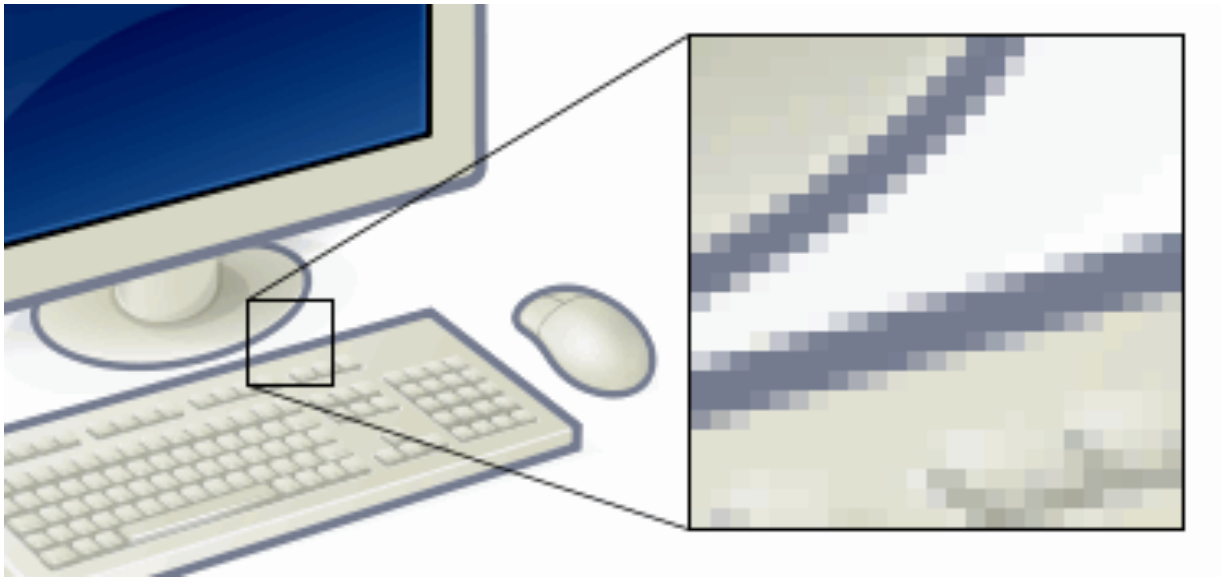
An image can be defined by a two-dimensional **array** specifically arranged in rows and columns.

Digital Image is composed of a finite number of elements, each of which elements have **a particular value at a particular location**. These elements are referred to as *picture elements, image elements, and pixels*. A *Pixel* is most widely used to denote the elements of a Digital Image.



- PPI

Pixels in Images



- 픽셀은 단색의 정사각형
- 전체 이미지의 크기를 표현할 때는 (세로픽셀수 x 가로픽셀수) 형식으로 표현
- **The term pixel is generally applied to 2D and is replaced by voxel in 3D images.**

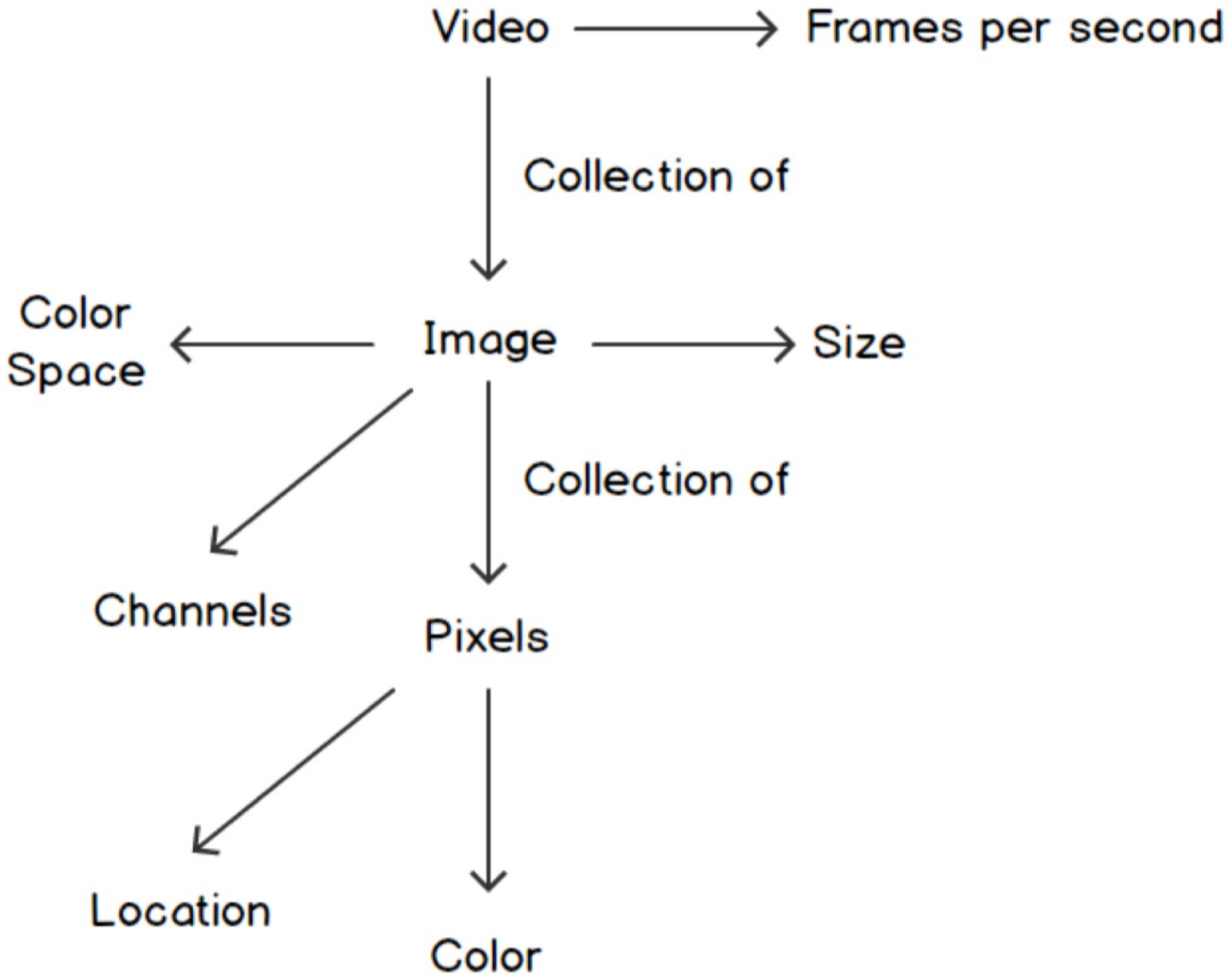
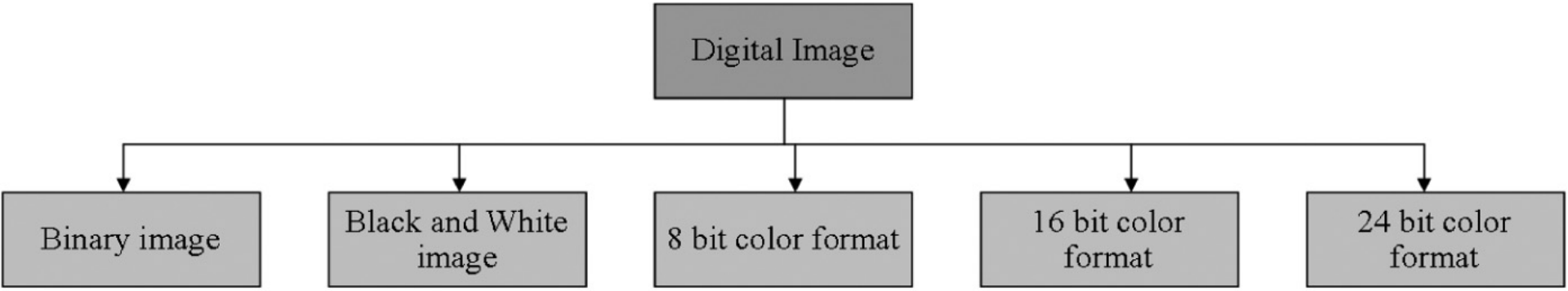

Video
Frames per second
Collection of
Color Space
Image
Size
Collection of
Channels
Pixels
Location
Color

Image category



■ Gray Scale

○ 모든 색이 흑백. 각 픽셀은 명도를 나타내는 숫자로 표현. 0은 검은색을 나타내고 숫자가 커질수록 명도가 증가하여 하얀색이 됨. 숫자는 보통 0~255의 8비트 부호없는 정수로 저장

■ RGB¶

○ Red, Green, Blue의 3가지 색의 명도를 뜻하는 숫자 3개가 합쳐진 벡터로 표현. 8비트 부호없는 정수를 사용하는 경우 (255, 0, 0)은 빨간색, (0, 255, 0)은 녹색, (0, 0, 255)는 파란색.

○ 픽셀 데이터가 스칼라가 아닌 벡터이므로 이미지 데이터는 (세로픽셀수 x 가로픽셀수) 형태의 2차원 배열로 표현하지 못하고 (세로픽셀수 x 가로픽셀수 x 색채널) 형태의 3차원 배열로 저장

■ HSV

○ Hue, Saturation, Value 세가지 값으로 표현

- 색상(Hue): 색상값 H는 가시광선 스펙트럼을 주파수 별로 고리모양으로 배치했을 때의 각도. 0°~360°의 범위를 갖고 360°와 0°는 빨강을 가리킴
- 채도(Saturation): 채도값 S는 특정한 색상의 진함의 정도. 가장 진한 상태를 100%이고 0%는 같은 명도의 무채색
- 명도(Value): 명도값 V는 밝은 정도를 나타냄. 순수한 흰색, 빨간색은 100%이고 검은색은 0%

○ HSV 색공간으로 표현된 파일은 imshow 명령으로 바로 볼 수 없음

- Matplotlib : rgb_to_hsv, hsv_to_rgb 상호변환

■ 색공간에 투명도(transparency)를 표현하는 A(Alpha) 채널이 추가된 RGBA, HSVA 등의 색공간도 있음

- BMP - Bitmap. This was probably the first type of digital image format that I can remember. Every picture on a computer seemed those days to be a BMP. In Windows XP the Paint program saves its images automatically in BMP. However, in Windows Vista and later images are now saved to JPEG. BMP is the basis platform for many other file types.

- JPG / JPEG - (Joint Photographic Experts Group) Jpeg format is used for color photographs, or any pictures wit h many blends or gradients. It is not good with sharp edges and tends to blur them a bit unless stored at high quality. This format became popular with the invention of the digital camera. Most, if not all, digital cameras d ownload photos to your computer as a Jpeg file. Obviously the digital camera manufacturers see the value in h igh quality images that ultimately take up less space.

- GIF - (Graphics Interchange Format) Gif format is best used for text, line drawings, screenshots, cartoons, and a nimations. Gif is limited to a total number of 256 colors or less, so Gif images are relatively small. It is common ly used for fast loading web pages. It also makes a great banner or logo for your web-page. Animated pictures can also be saved in GIF format as a sequence of static images. For example, a flashing banner would be saved as a Gif file.

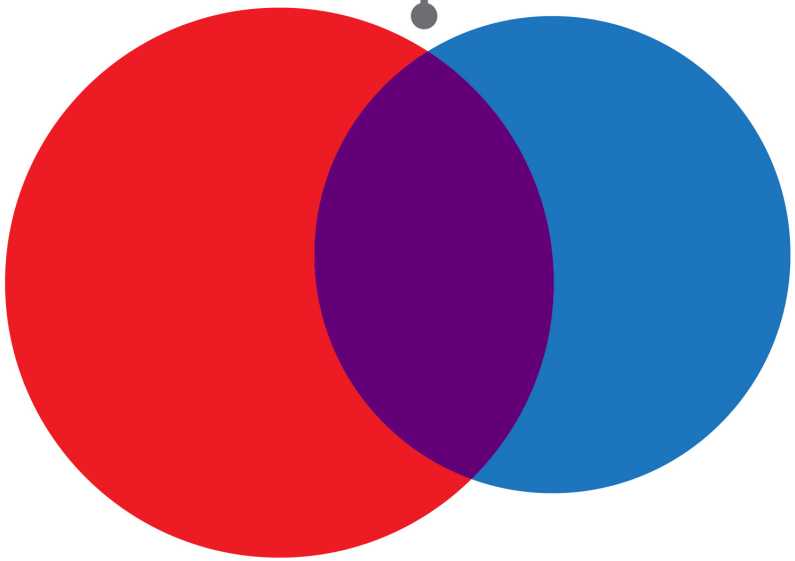
- PNG - (Portable Networks Graphic) This lossless formats is one of the best image formats. It was not always co mpatible with all web browsers or image software, but nowadays it is the best image format to use for website . I use .png for logos and screenshots. One of its most astonishing abilities is being able to compress images lo sslessly (without loss of pixels), although the final compressed size varies between image editors.

- TIFF - (Tagged Image File Format) This file format has not been updated since 1992 and is now owned by Ado be. It can store an image and data (tags) in the one file. TIFF can be compressed, but it is rather its ability to st ore image data in a lossless format that makes a TIFF file a useful image archive, because unlike standard JPEG files, a TIFF file using lossless compression (or none) may be edited and re-saved without losing image quality. This file is commonly used for scanning, faxing, word processing, and so on. It is no longer a common file form at to use with your digital photos, as jpeg is great quality and takes up less space.

이미지 파일 형식

	BMP	JPG	PNG	GIF
Compressed	FALSE	TRUE	TRUE	TRUE
Lossless	TRUE	FALSE	TRUE	TRUE
Transparency	FALSE	FALSE	TRUE	TRUE
Translucency	FALSE	FALSE	TRUE	FALSE
Recommended for photographs	FALSE	TRUE	FALSE	FALSE
Recommended for static graphics/icons	FALSE	FALSE	TRUE	FALSE
Recommended for animated graphics/icons	FALSE	FALSE	FALSE	TRUE

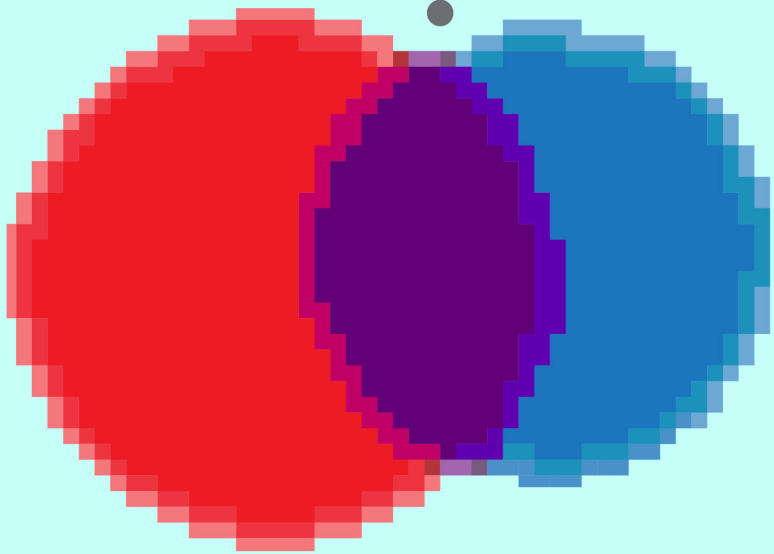
Vector



AI EPS CGM

PDF SVG CDR

Raster



BMP TIFF PCX

GIF PNG JPEG

Image Processing

Digital Image Processing means processing digital image by means of a digital computer. We can also say that **it is a use of computer algorithms**, in order to get enhanced image to extract some useful information.

Image processing mainly include the following steps:

1. Importing the image via image acquisition tools
2. Analysing and manipulating the image
3. Output in which result can be altered image or a report which is based on analysing that image.

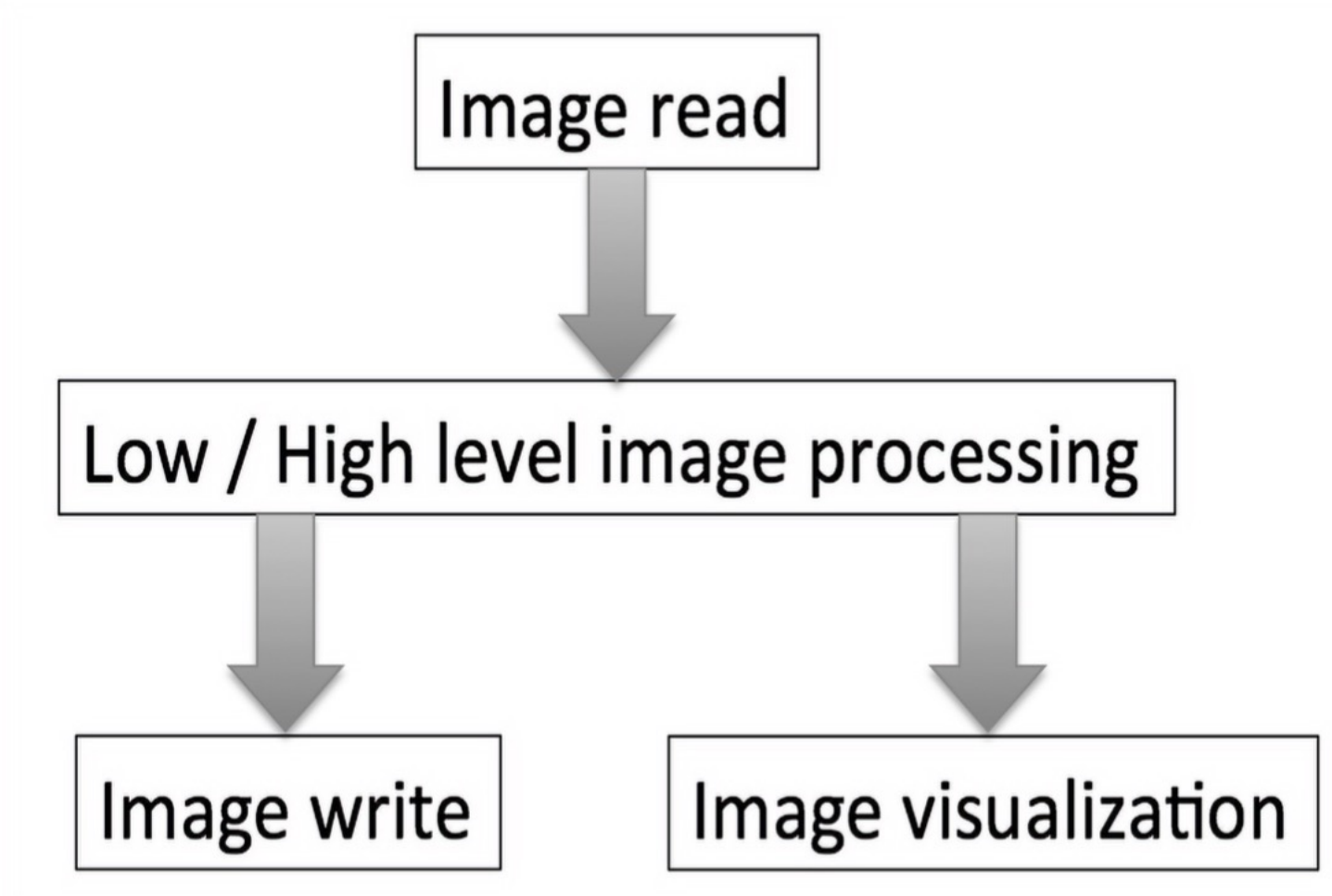


Image read
Low / High level image processing
Image write
Image visualization

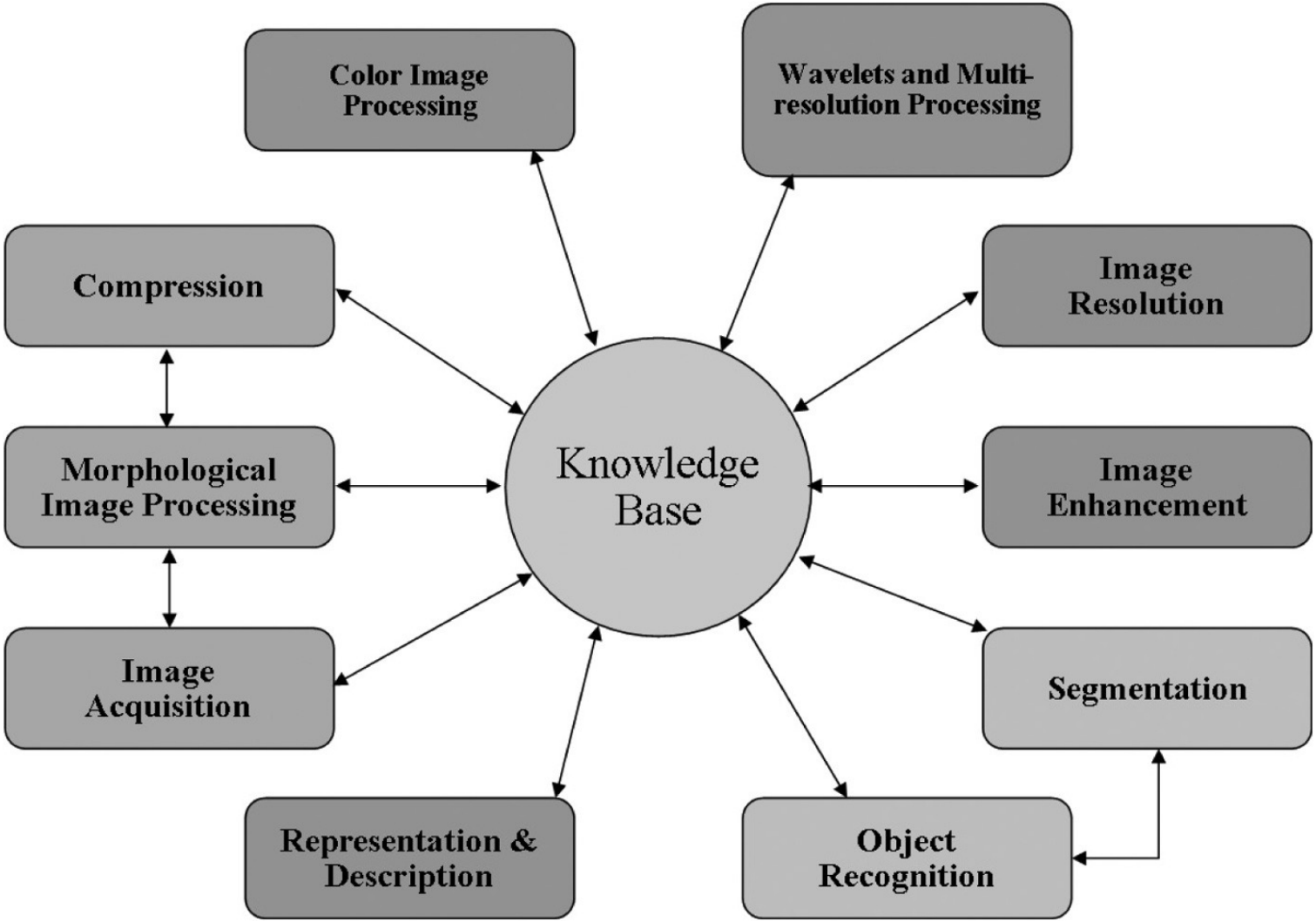
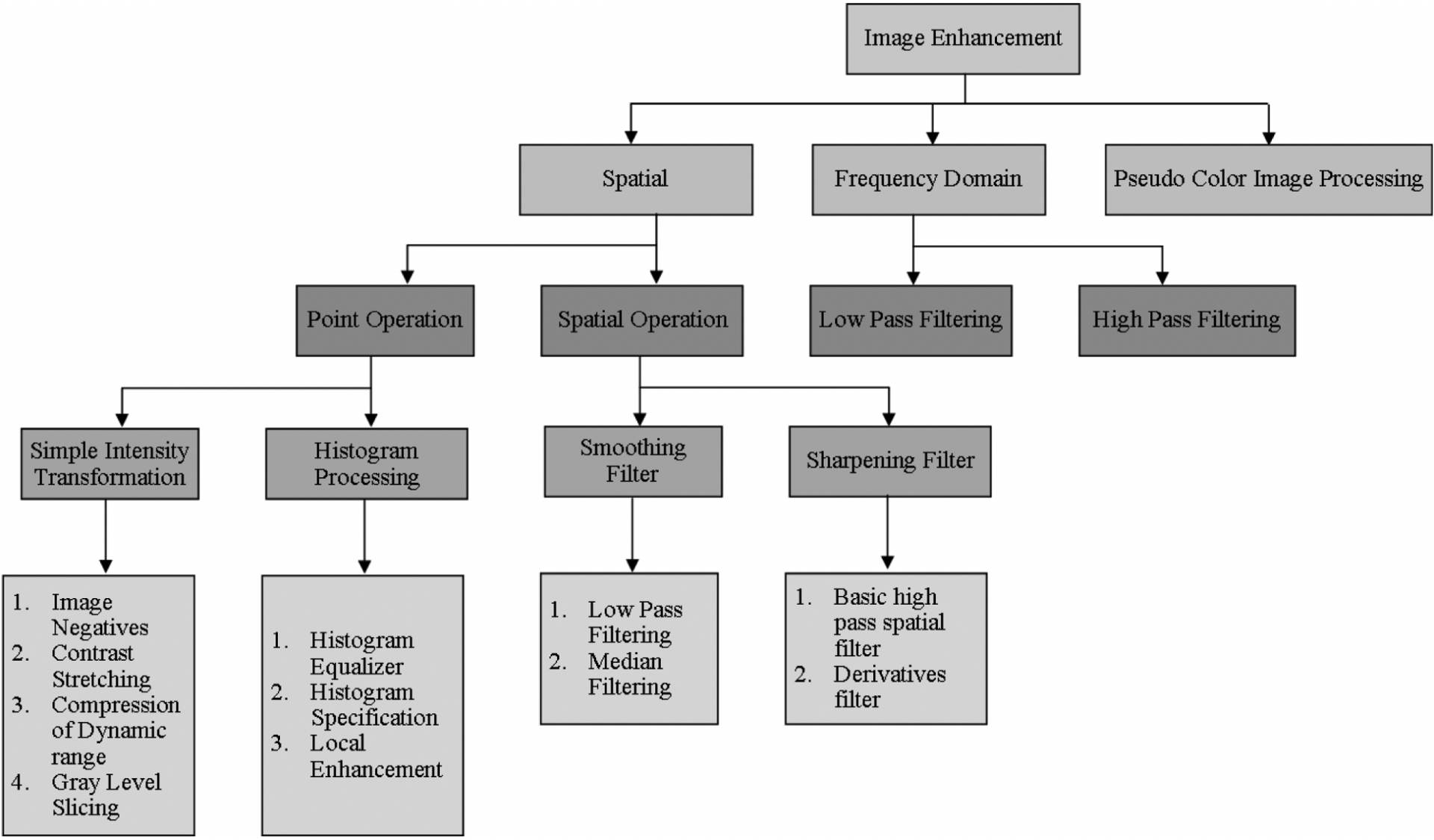

Color Image Processing
Wavelets and Multi-resolution Processing
Compression
Image Resolution
Morphological Image Processing
Knowledge Base
Image Enhancement
Image Acquisition
Segmentation
Representation & Description
Object Recognition

Image Enhancement Field



No Free Lunch

1. Our model is a simplification of **reality**

2. Simplification is based on assumptions (model bias)

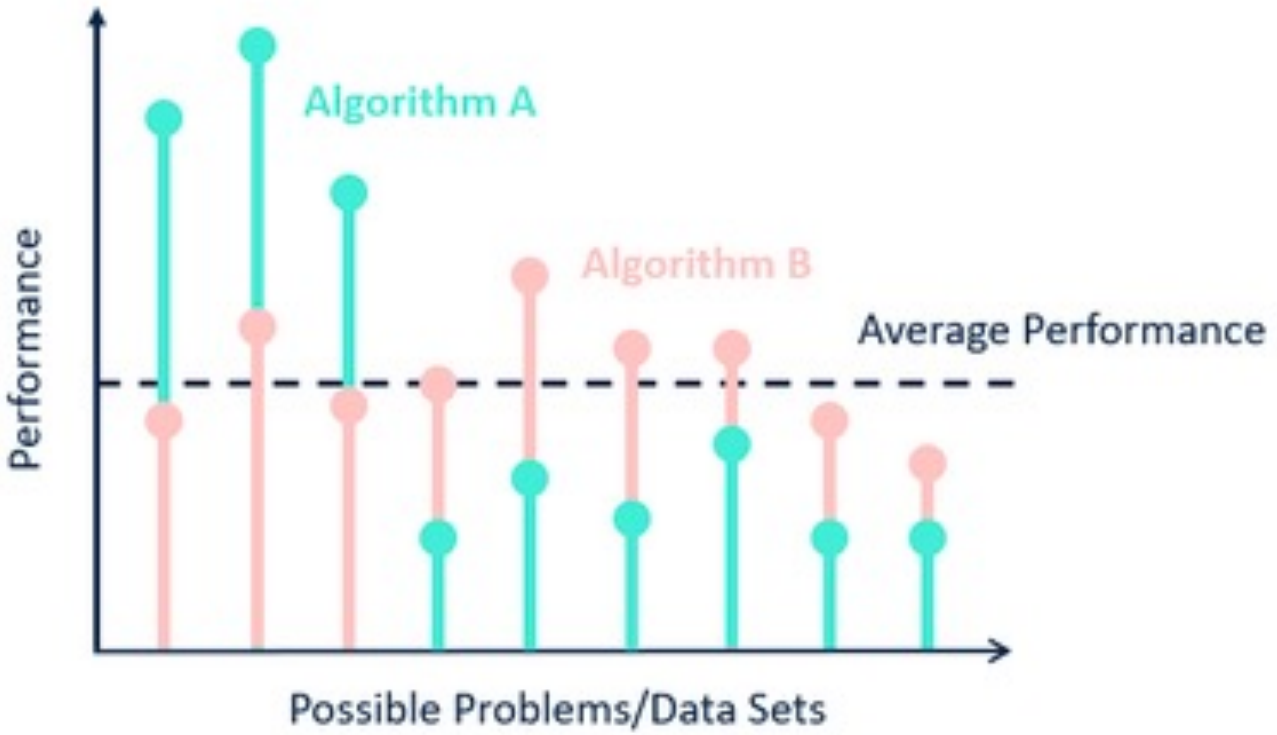
3. Assumptions fail in certain situations

"No one model works best for all possible situations."

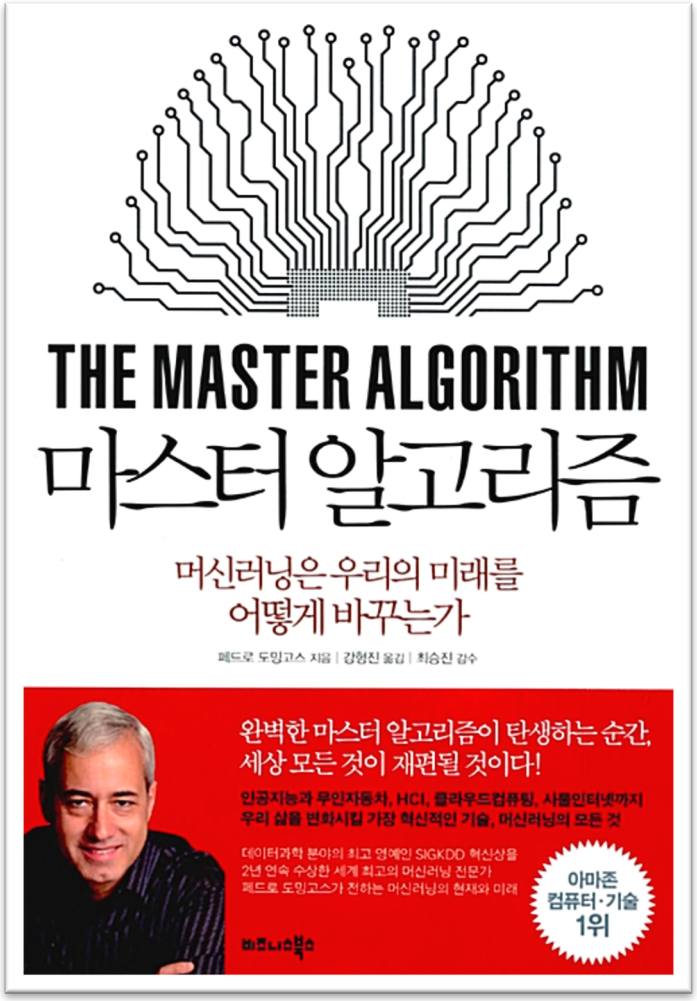
The **supervised** learning no-free-lunch theorems. (2002)

D. H. Wolpert.

No Free Lunch Theorems for **Optimizations** (1997)



어떤 특정 정책에 의해 얼핏 보면 이득을 얻는 것 같지만,
그것은 한 측면의 이득일 뿐이고 반드시 이면에 다른 측면이 있고, 그 측면에서 손해가 발생



A Few Useful Things to Know about Machine Learning

Pedro Domingos
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
pedrod@cs.washington.edu

ABSTRACT

Machine learning algorithms can figure out how to perform important tasks by generalizing from examples. This is often feasible and cost-effective where manual programming is not. As more data becomes available, more ambitious problems can be tackled. As a result, machine learning is widely used in computer science and other fields. However, developing successful machine learning applications requires a substantial amount of "black art" that is hard to find in textbooks. This article summarizes twelve key lessons that machine learning researchers and practitioners have learned. These include pitfalls to avoid, important issues to focus on, and answers to common questions.

1. INTRODUCTION

Machine learning systems automatically learn programs from data. This is often a very attractive alternative to manually constructing them, and in the last decade the use of machine learning has spread rapidly throughout computer science and beyond. Machine learning is used in Web search, spam filters, recommender systems, ad placement, credit scoring, fraud detection, stock trading, drug design, and many other applications. A recent report from the McKinsey Global Institute asserts that machine learning (a.k.a. data mining or predictive analytics) will be the driver of the next big wave of innovation [16]. Several fine textbooks are available to interested practitioners and researchers (e.g. [17, 25]). However, much of the "folk knowledge" that is needed to successfully develop machine learning applications is not readily available in them. As a result, many machine learning projects take much longer than necessary or wind up producing less-than-ideal results. Yet much of this folk knowledge is fairly easy to communicate. This is the purpose of this article.

Many different types of machine learning exist, but for illustration purposes I will focus on the most mature and widely used one: classification. Nevertheless, the issues I will discuss apply across all of machine learning. A *classifier* is a system that inputs (typically) a vector of discrete and/or continuous *feature values* and outputs a single discrete value, the *class*. For example, a spam filter classifies email messages into "spam" or "not spam," and its input may be a Boolean vector $\mathbf{x} = (x_1, \ldots, x_j, \ldots, x_d)$, where $x_j = 1$ if the jth word in the dictionary appears in the email and $x_j = 0$ otherwise. A *learner* inputs a *training set* of *examples* $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,d})$ is an observed input and y_i is the corresponding output, and outputs a classifier. The test of the learner is whether this classifier produces the correct output y_t for future examples $\mathbf{x}_t$ (e.g., whether the spam filter correctly classifies previously unseen emails as spam or not spam).

2. LEARNING = REPRESENTATION + EVALUATION + OPTIMIZATION

Suppose you have an application that you think machine learning might be good for. The first problem facing you is the bewildering variety of learning algorithms available. Which one to use? There are literally thousands available, and hundreds more are published each year. The key to not getting lost in this huge space is to realize that it consists of combinations of just three components. The components are:

Representation. A classifier must be represented in some formal language that the computer can handle. Conversely, choosing a representation for a learner is tantamount to choosing the set of classifiers that it can possibly learn. This set is called the *hypothesis space* of the learner. If a classifier is not in the hypothesis space, it cannot be learned. A related question, which we will address in a later section, is how to represent the input, i.e., what features to use.

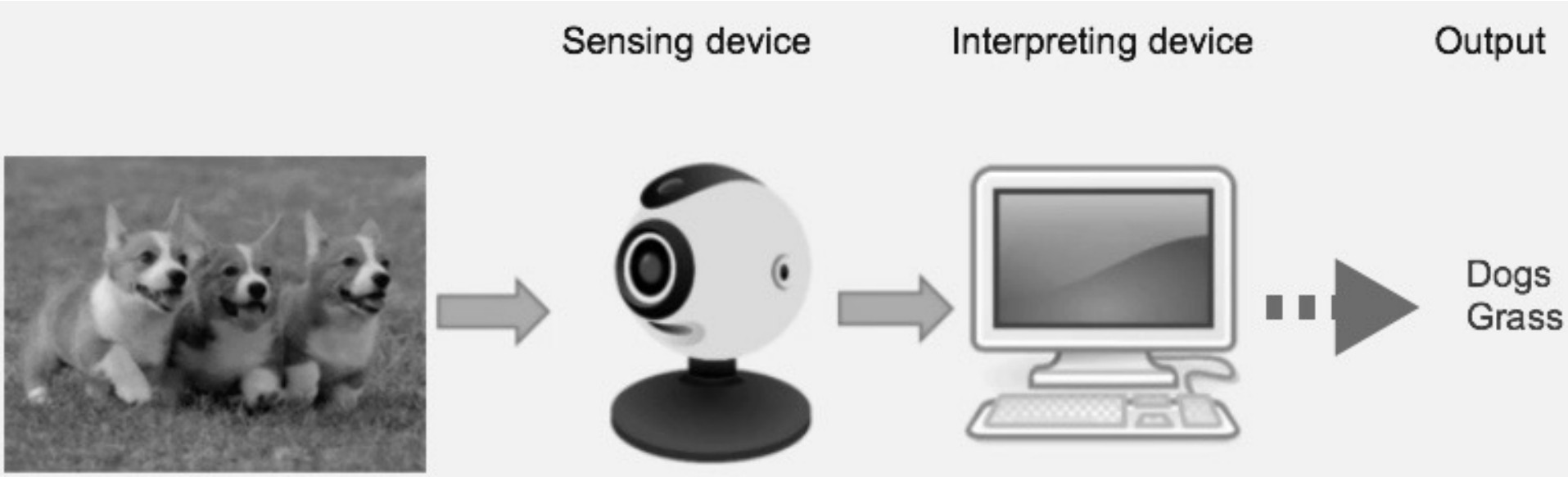
Evaluation. An evaluation function (also called *objective function* or *scoring function)* is needed to distinguish good classifiers from bad ones. The evaluation function used internally by the algorithm may differ from the external one that we want the classifier to optimize, for ease of optimization (see below) and due to the issues discussed in the next section.

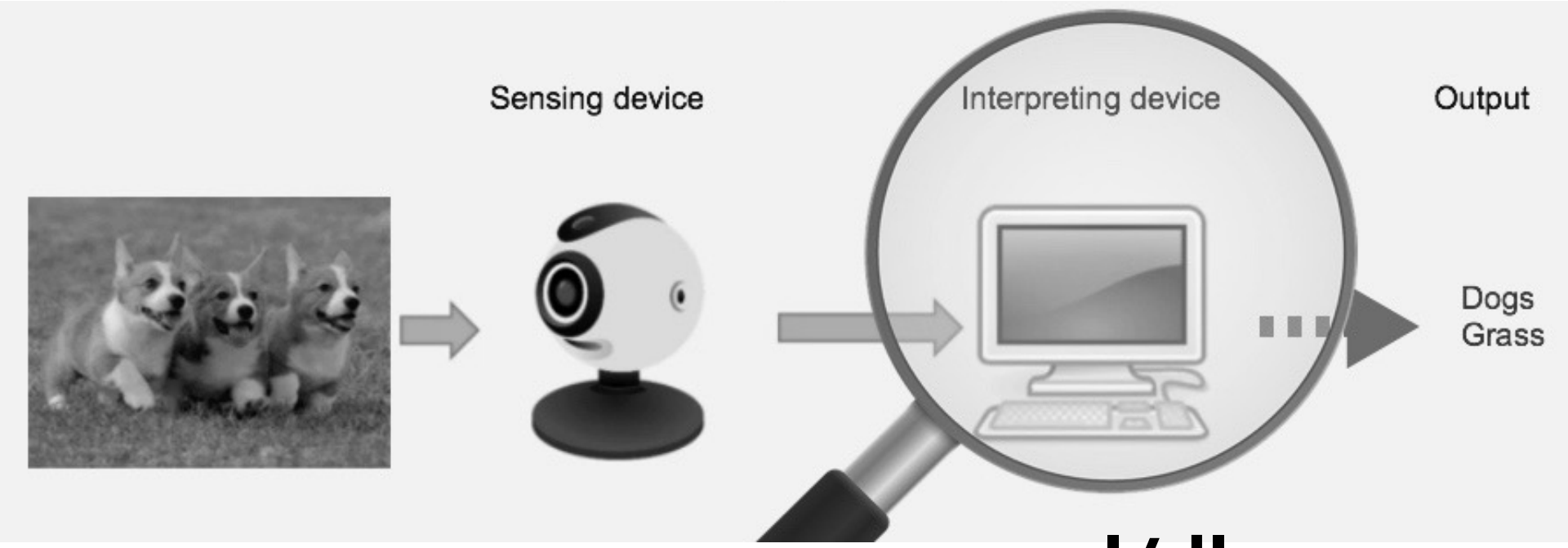
Optimization. Finally, we need a method to search among the classifiers in the language for the highest-scoring one. The choice of optimization technique is key to the efficiency of the learner, and also helps determine the classifier produced if the evaluation function has more than one optimum. It is common for new learners to start out using off-the-shelf optimizers, which are later replaced by custom-designed ones.

Table 1 shows common examples of each of these three components. For example, k-nearest neighbor classifies a test example by finding the k most similar training examples and predicting the majority class among them. Hyperplane-based methods form a linear combination of the features per class and predict the class with the highest-valued combination. Decision trees test one feature at each internal node,

(sensing device responsible for capturing images of the environment)

(Interpreting device responsible for understanding the image content)

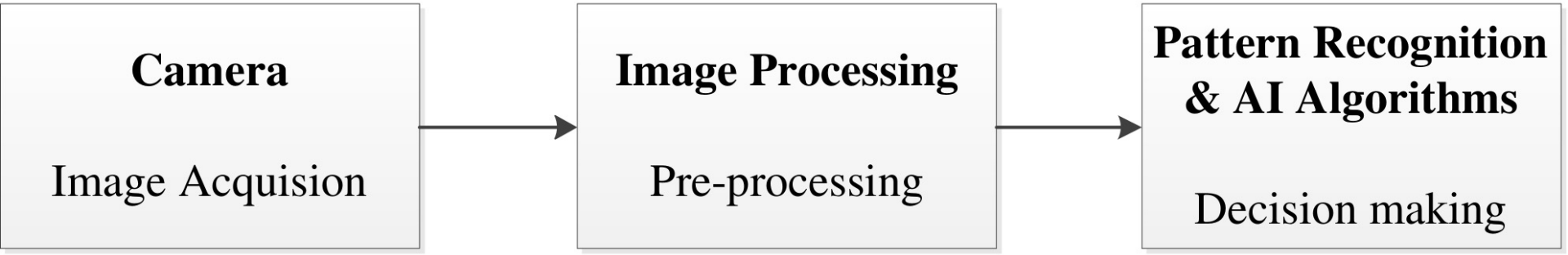

Sensing device
Interpreting device
Output
Dogs
Grass



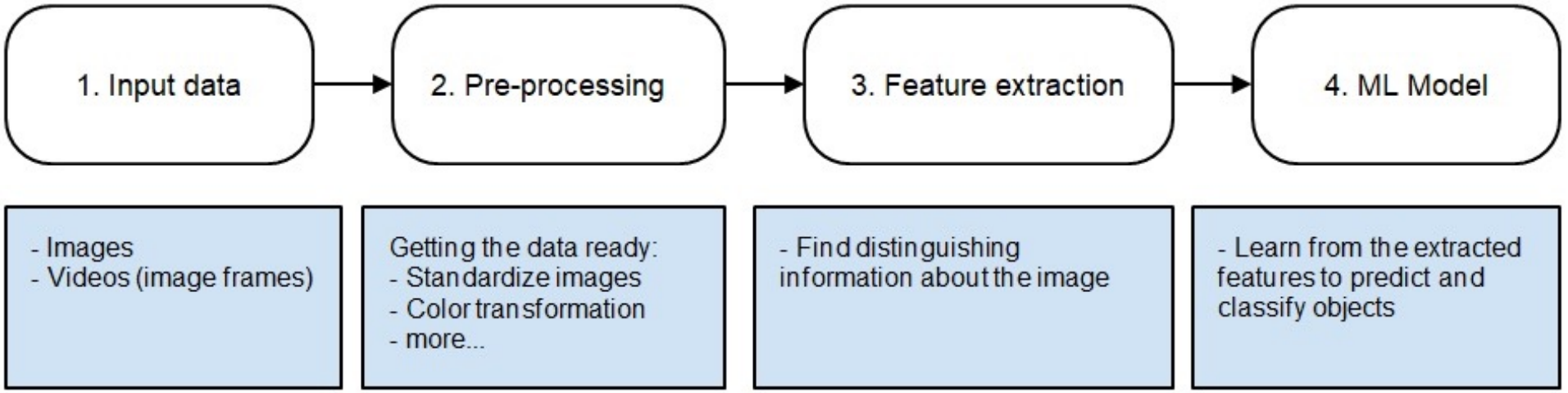
ml/dl

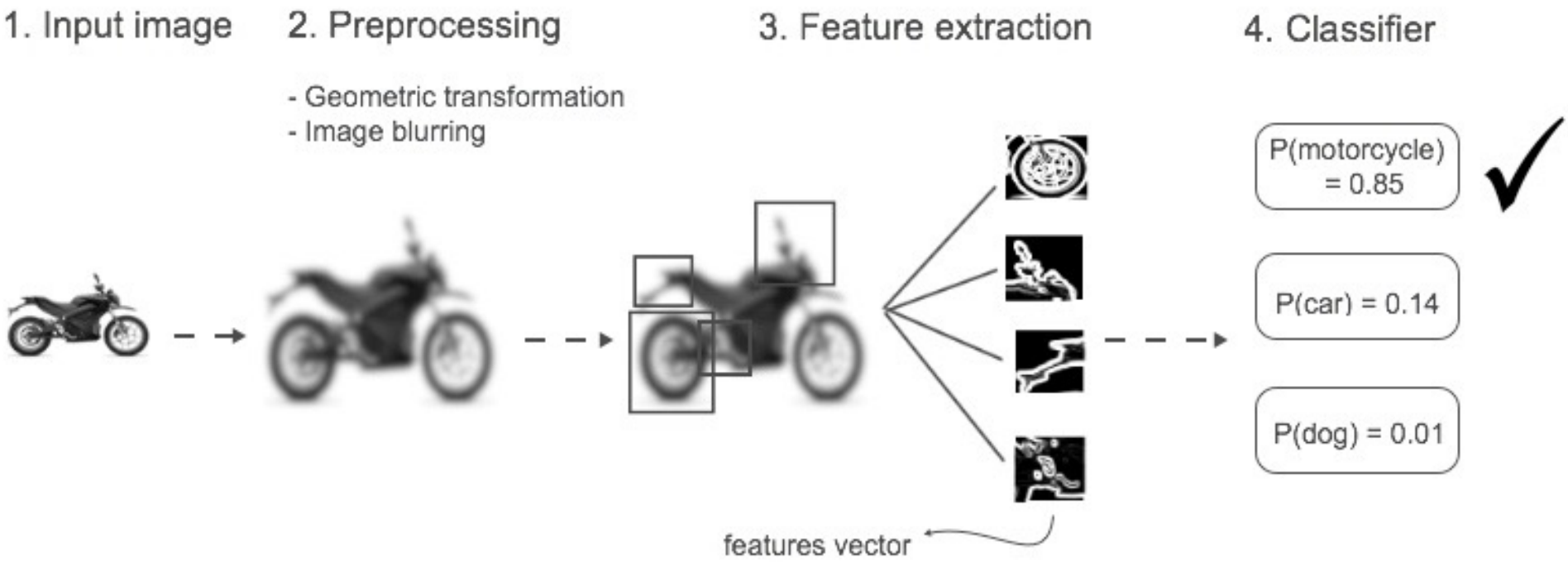

Seeing
Processing
Intelligent Decision

(a)


Camera
Image Acquision
Image Processing
Pre-processing
Pattern Recognition & AI Algorithms
Decision making

(b)


1. Input data
2. Pre-processing
3. Feature extraction
4. ML Model
- Images
- Videos (image frames)
Getting the data ready:
- Standardize images
- Color transformation
- more...
- Find distinguishing information about the image
- Learn from the extracted features to predict and classify objects


1. Input image
2. Preprocessing
- Geometric transformation
- Image blurring
3. Feature extraction
4. Classifier
P(motorcycle) = 0.85
P(car) = 0.14
P(dog) = 0.01
features vector

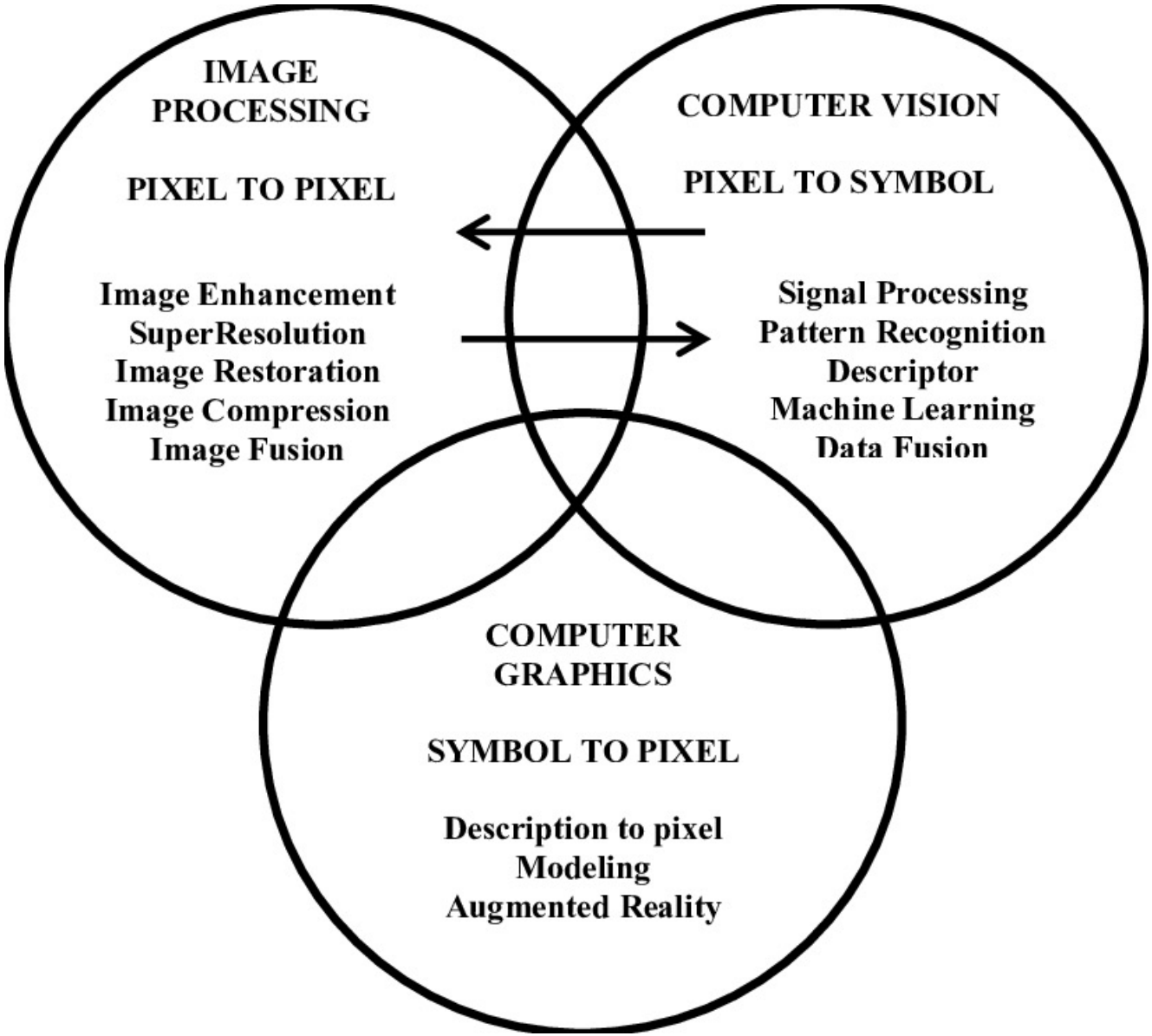


IMAGE PROCESSING
PIXEL TO PIXEL
Image Enhancement
SuperResolution
Image Restoration
Image Compression
Image Fusion
COMPUTER VISION
PIXEL TO SYMBOL
Signal Processing
Pattern Recognition
Descriptor
Machine Learning
Data Fusion
COMPUTER GRAPHICS
SYMBOL TO PIXEL
Description to pixel
Modeling
Augmented Reality

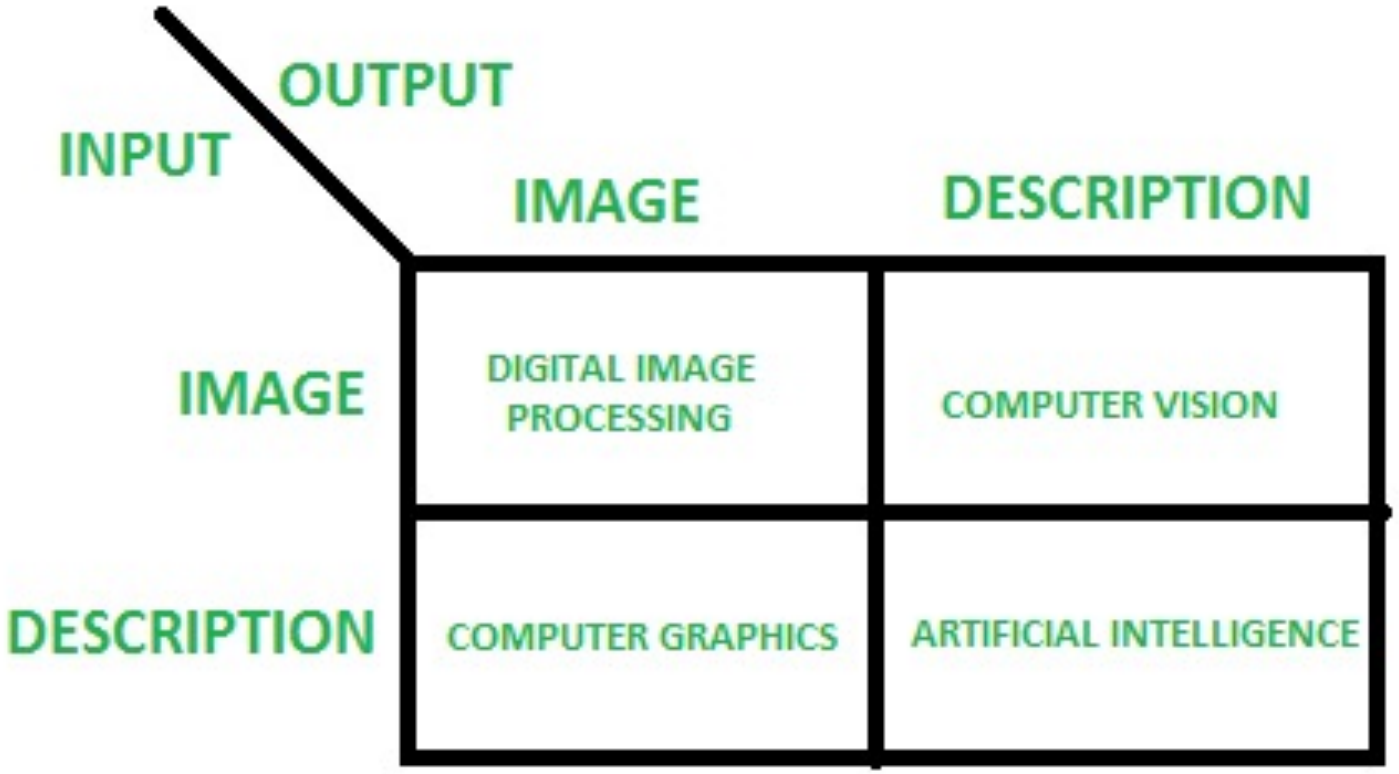

OUTPUT
INPUT
IMAGE
DESCRIPTION
IMAGE
DIGITAL IMAGE PROCESSING
COMPUTER VISION
DESCRIPTION
COMPUTER GRAPHICS
ARTIFICIAL INTELLIGENCE

Image processing is the process of creating a new image from an existing image, typically simplifying or enhancing the content in some way. It is a type of digital signal processing and is not concerned with understanding the content of an image.

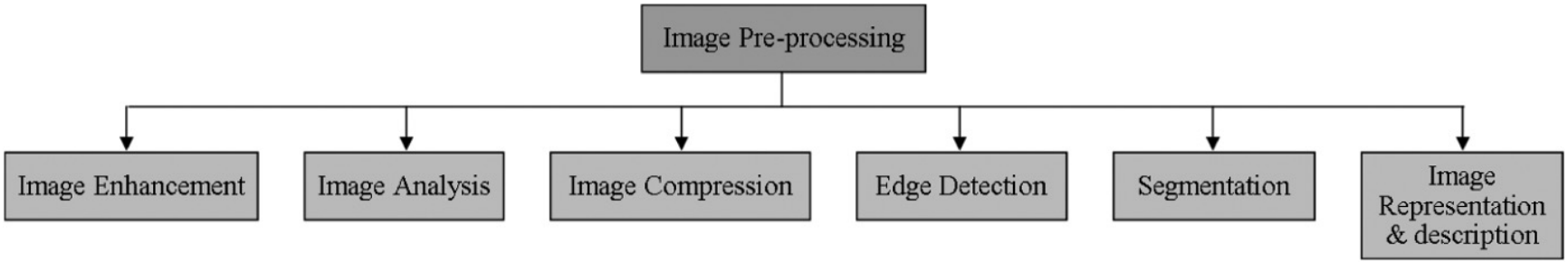
A given computer vision system may require image processing to be applied to raw input, e.g. pre-processing images.

Examples of image processing include:

- Normalizing photometric properties of the image, such as brightness or color.
- Cropping the bounds of the image, such as centering an object in a photograph.
- Removing digital noise from an image, such as digital artifacts from low light levels.

Image Processing for ML/DL

■ Transitional Image Pre-processing



■ Essential

○ Uniform Image Size

■ Techniques to improve performance of machine learning models

○ Uniform Aspect Ratio

○ Normalized Image Inputs

○ Uniform Image Size

○ Dimensionality Reduction

- ➢ Convert color images to grayscale to reduce computation complexity

○ Mean and Perturbed Images

- ➢ Standardize images

○ **Data Augmentation**

Data augmentation

Original image

De-texturized

De-colorized

Edge enhanced

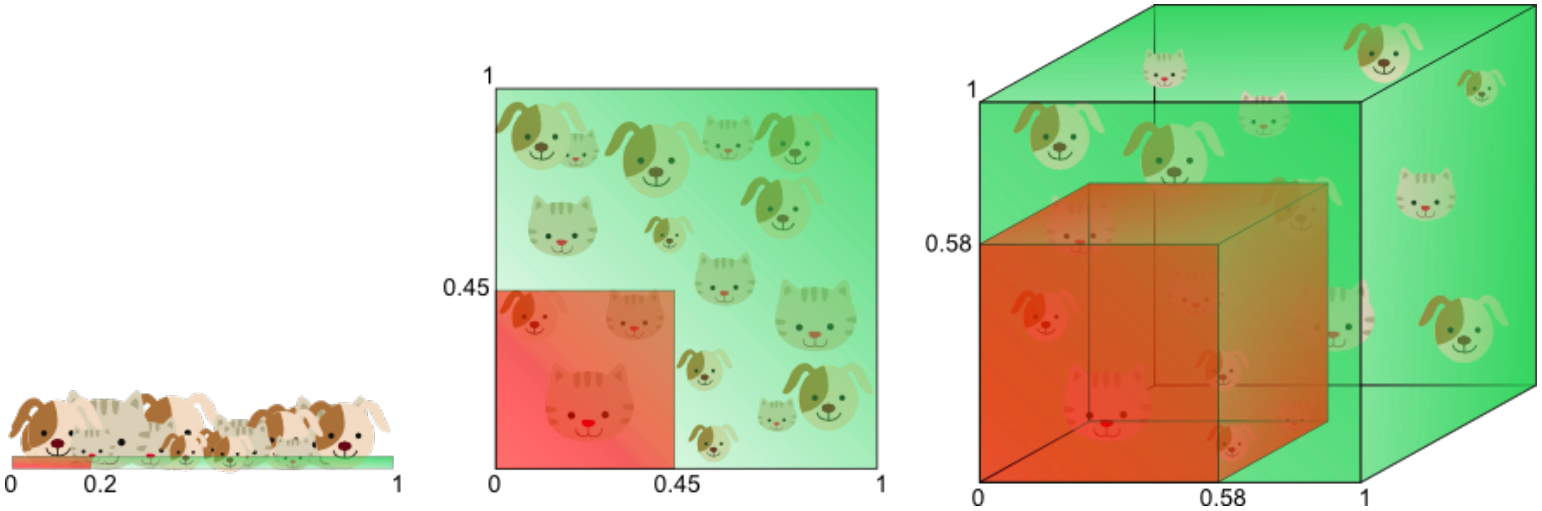
Salient edge map

Flip/rotate

Why use features?

데이터의 차원이 증가할수록
학습에 필요한 데이터의 개수가 기하급수적으로 늘어나는 문제

- 같은 비율의 공간을 모델링하기 위해서 필요한 데이터 량이 급속도록 증가
- Example: 공간의 20%를 채우기 위해서 필요한 데이터 수
- 0.2(1차원) < 0.9(2차원) < 1.74 (3차원)



https://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/

■ The Curse of Dimensionality

○ 저차원에서는 충분했던 데이터 양도 고차원이 되면 공간(학습 모델)을 설명하기에 부족해 질 수 있음

○ 모델을 설명하기에 데이터가 부족 > Overfitting문제

■ 해결책

○ 데이터 양

○ 중요한 특징만 뽑자

- ➢ Dimensionality Reduction
- ➢ Feature Extraction
- ➢ Feature Selection

○ 사전지식 이용 > Regularziation

충분성: 충분하지 않는 데이터
대표성: sampling noise : 우연에 의한 대표성 없는 데이터
sampling bias : 큰 샘플도 표본추출방법이 잘못되면 대표성을 띠지 못함

[데이터 품질]
불완전성: 데이터에 특성이 없거나 값이 누락
노이즈가 많은: 데이터에 잘못된 레코드(손상된) 또는 이상값이 있음
불일치: 데이터에 충돌하는 레코드 또는 일치하지 않는 값

Garbage in, Garbage out

- 우수한 예측 모델을 구축하려면 우수한 데이터가 필요
- 데이터 품질을 높여서 궁극적으로 모델 성능을 높이기 위해서 데이터 상태 검사를 수행하여 조기에 데이터 문제를 발견하고 적절한 데이터 처리 및 정리 단계를 결정하는 것이 중요

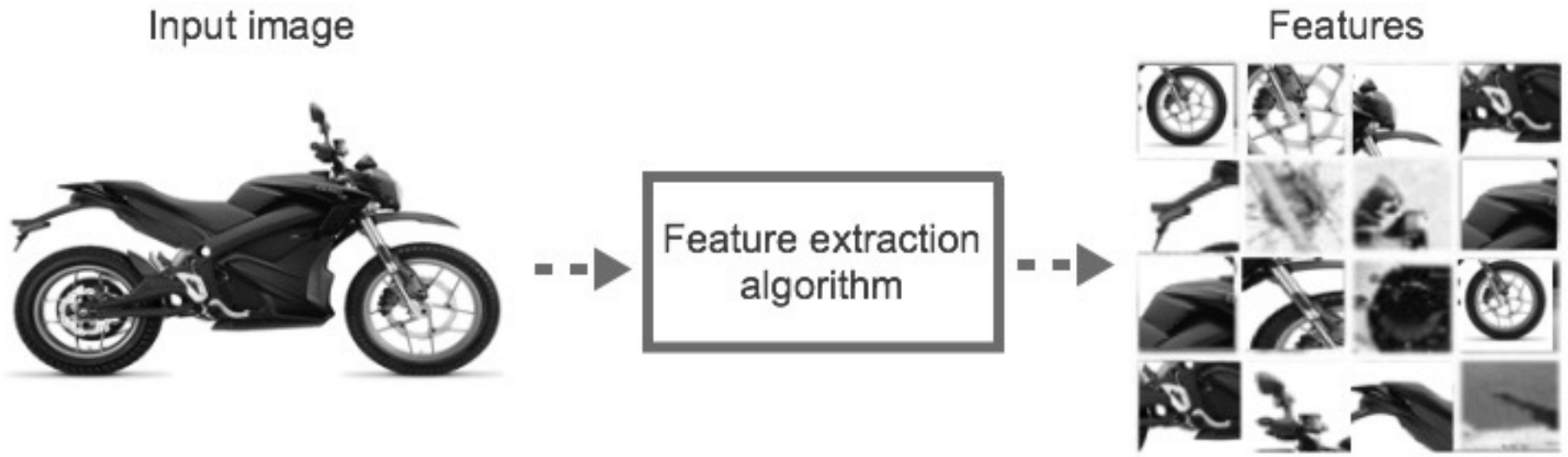
Data Cleaning

"In our experience, the tasks of exploratory data mining and data cleaning constitute 80% of the effort that determines 80% of the value of the ultimate data."

T. Dasu and T. Johnson
Authors of *Exploratory Data Mining and Data Cleaning*

- Good data scientists spend most of their time cleaning and formatting data.
- The rest spend most of their time complaining there is no data available.
- Data munging or data wrangling is the art of acquiring data and preparing (cleaning) it for analysis.

feature is a measurable piece of data in your image that is unique to this specific object. It maybe a distinct color in an image or a specific shape such as a line, edge, or an image segment.



features vector that is a array that makes a robust representation of the object.

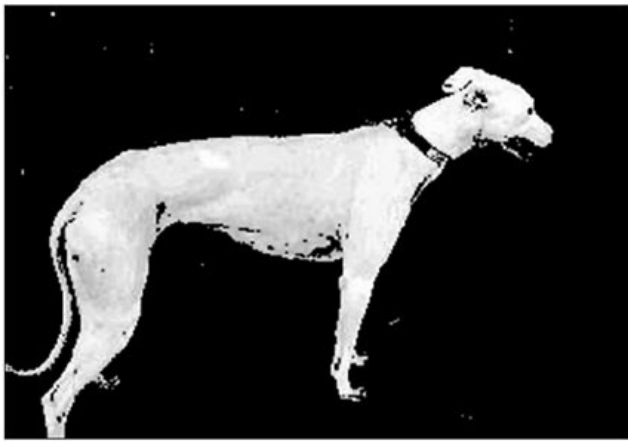
Speciailly Image

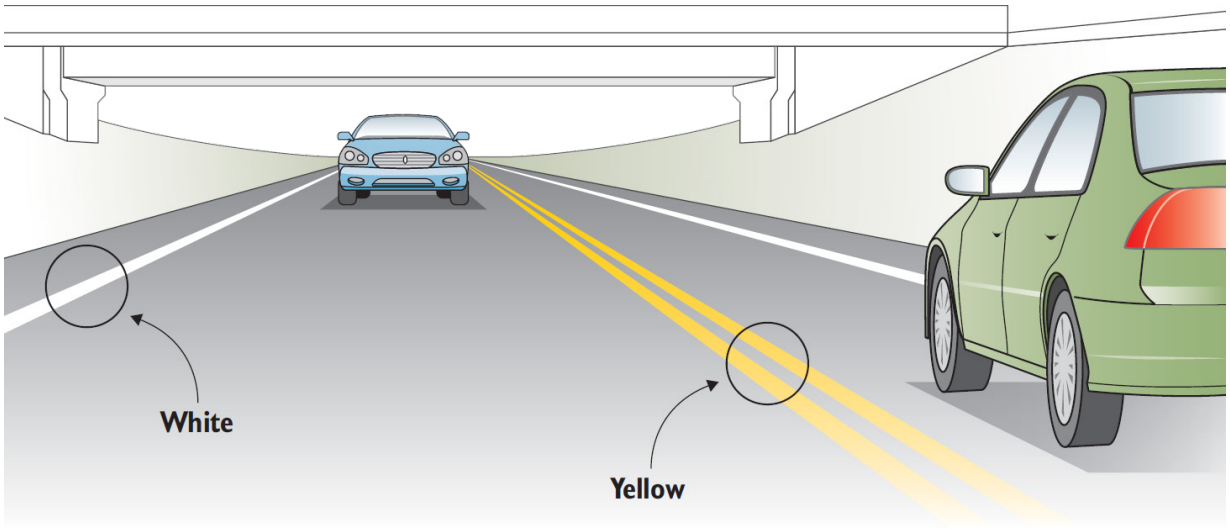
Local features

such as edge and keypoints.

Global features

such as color histogram and pixel count. By global, it is meant that the feature describes the entire image.




White
Yellow

Raw data	Featured Data
No information loss → Discover Unrecognized patterns 	Some information loss → Degrade performance sometime 
Curse of dimensionality 	Dimensionality Reduction 
Higher computational cost → Slower learning time 	Lower computation cost → Faster learning time 
Large-scale storage is required 	Small-scale storage is fine 

images, even small ones, have a very large number of pixels, where each pixel is applied as an input to the model. For a grayscale image of size 100×100 pixels, there are 100×100 = 10,000 input variables to be applied to the model. For a small dataset of 100 samples, there will be a total of 100×10,000 = 1,000,000 inputs across the entire dataset. If the image is Red-Green-Blue (RGB), the total number is multiplied by 3. This requires a large memory in addition to being computationally intensive.

Another reason why feature extraction is preferred before training is that the input image has different types of objects with different properties, and **we just want to target a single object**.

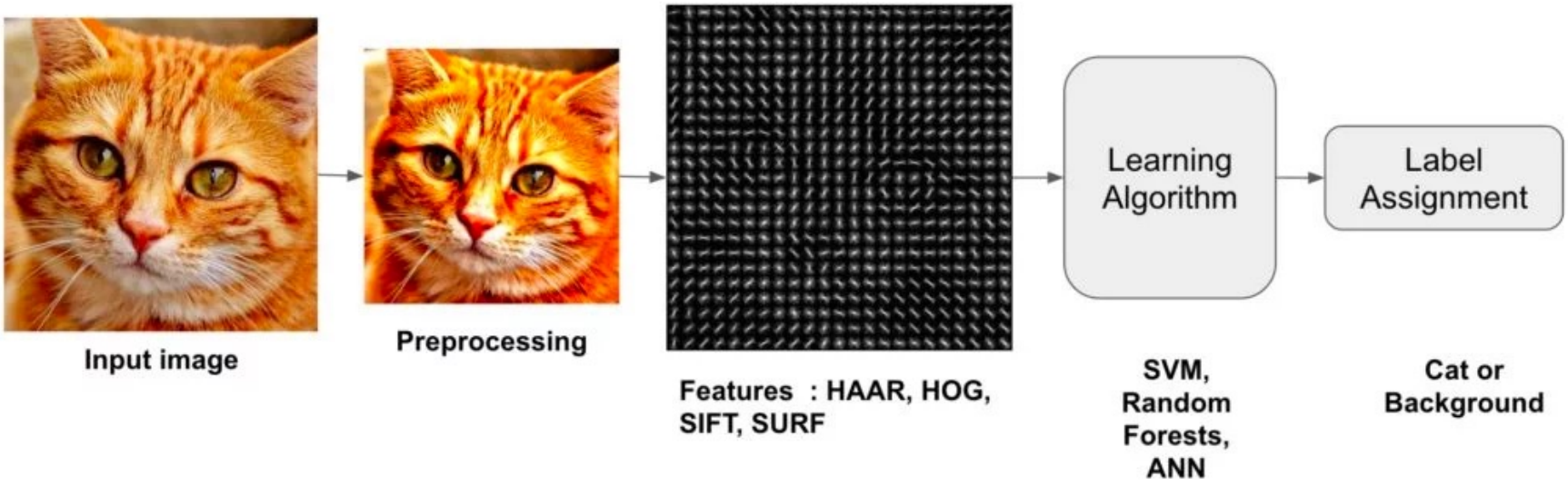
repeatability

Feature after looking at one image

Feature after looking at 1000s of images

A good feature will help us recognize an object in all the ways it may appear. Characteristics of a good feature:

- Identifiable
- Easily tracked and compared
- Consistent across different scales, lighting conditions, and viewing angles
- Still visible in noisy images or when only part of an object is visible



- Histogram of Oriented Gradients (HOG)
- Haar Cascades
- Scale-Invariant Feature Transform (SIFT)
- Speeded Up Robust Feature (SURF)

- Color Histogram
- Edge
- Texture
  - GLCM
  - GLGCM
  - LBP

Correlation

measurement of the similarity between two signals/sequences.

between two signals

Convolution

measurement of effect of one signal on the other signal.

between signal and a system(filter)

Theoretically, convolution are **linear operations** on the signal or signal modifiers, whereas correlation is a measure of similarity between two signals. As you rightly mentioned, the basic difference between convolution and correlation is that the convolution process rotates **the matrix by 180 degrees**..

linear systems theory : linear operation result > linear

Translational Invariance

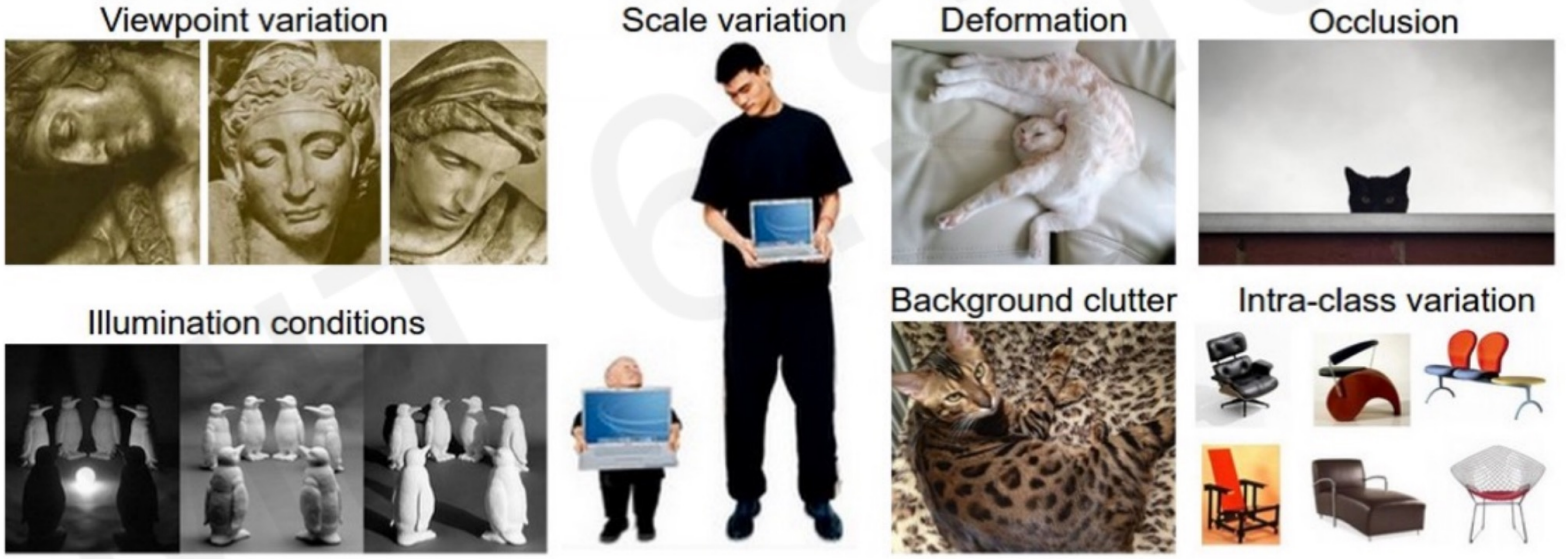
Our vision system should be to sense, respond or detect the same object regardless of where it appears in the image.

Locality

Our vision system focusses on the local regions, without regard to what else is happening in other parts of the image.

Manul Feature Extraction

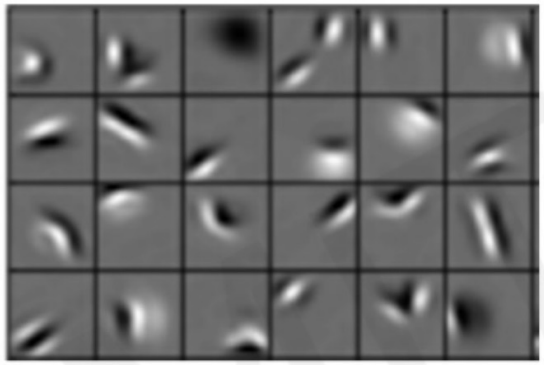
Domain knowledge → Define features → Detect features to classify



Learning Feature Representation

Can we learn a **hierarchy of features** directly from the data instead of hand engineering?

Low level features



Edges, dark spots

Mid level features

Eyes, ears, nose

High level features

Facial structure

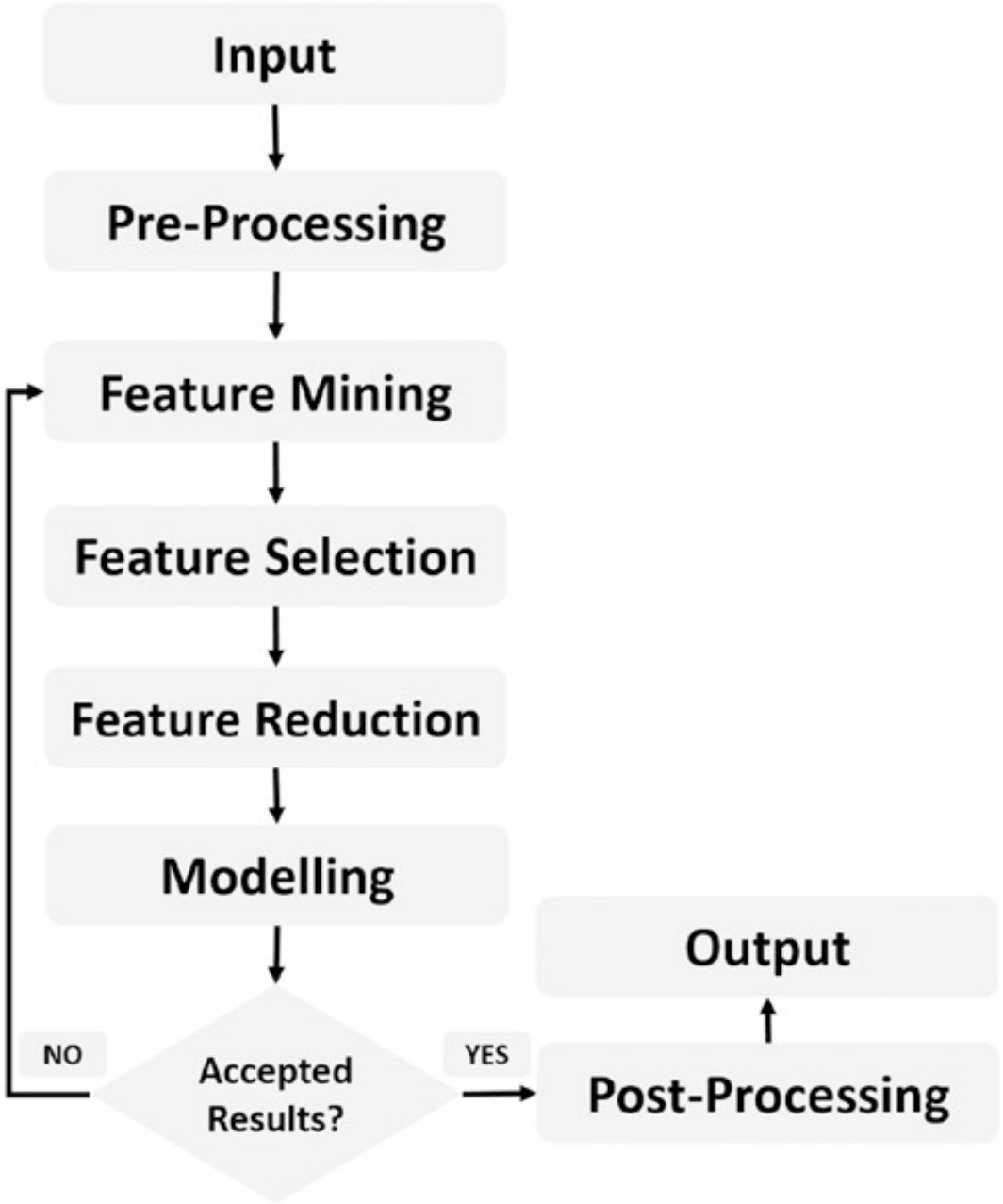
Why Deep Learning?

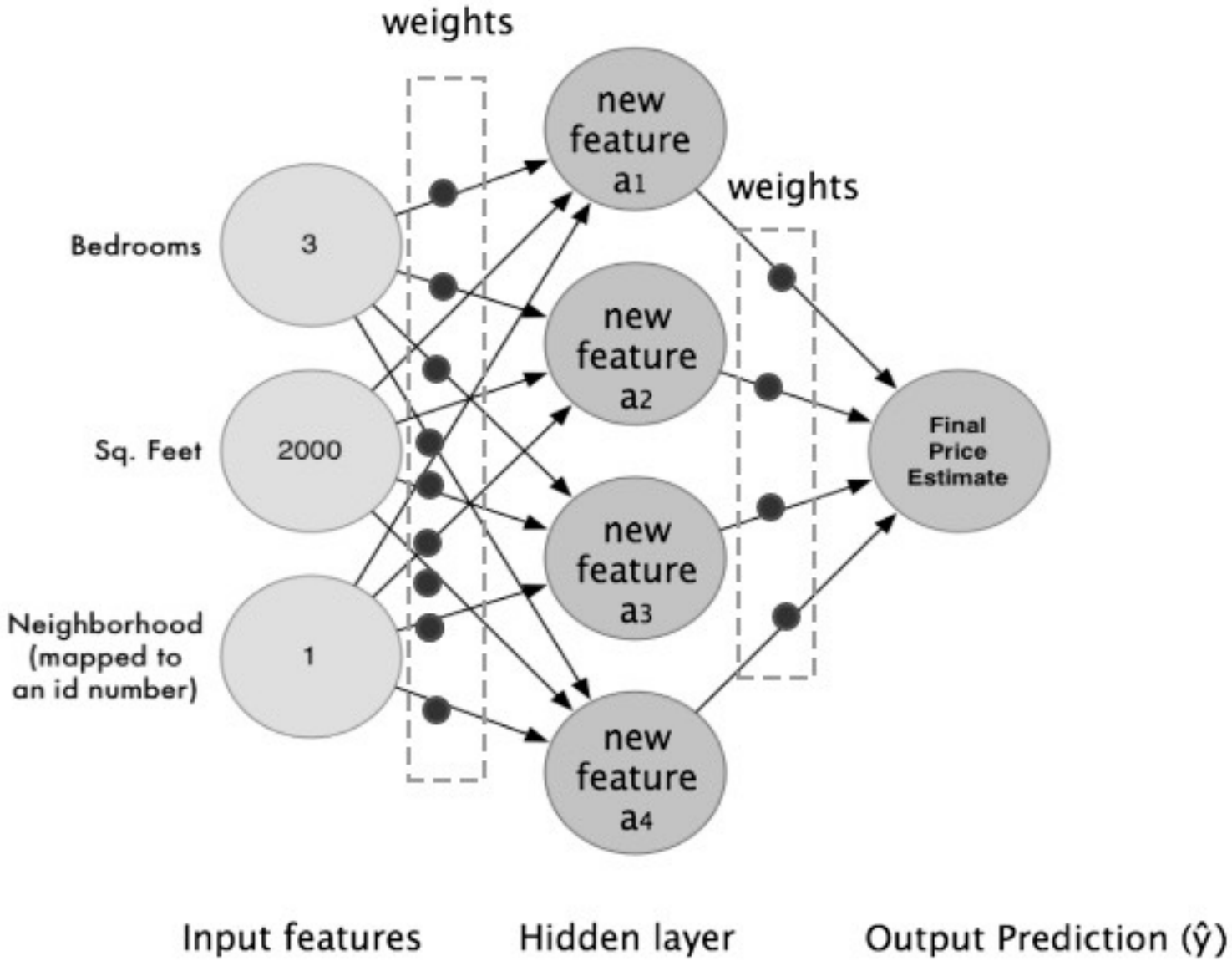
Beginning Application Development with TensorFlow and Keras

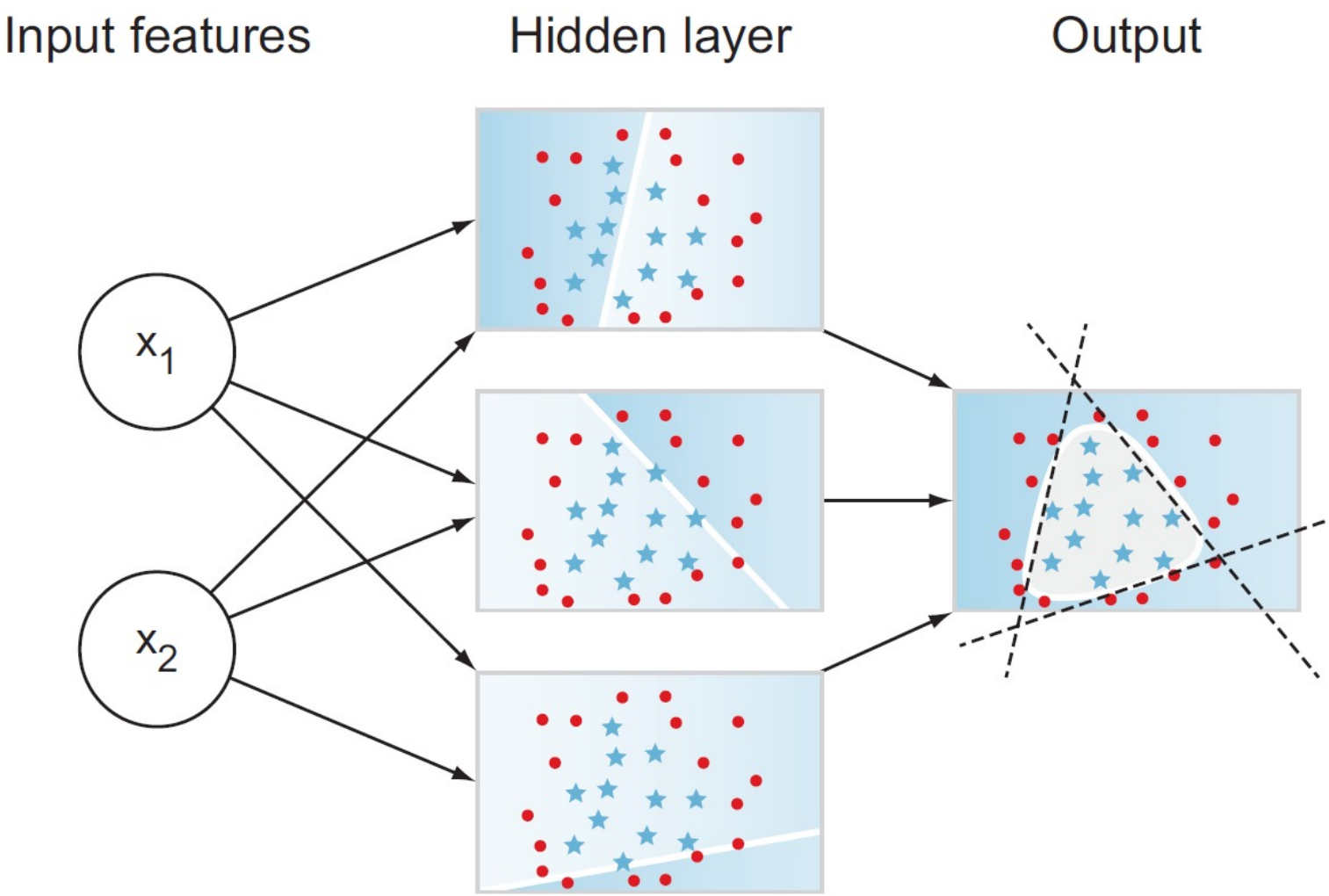
3-5

https://brunch.co.kr/@itschloe1/8

http://hyperparameter.space/blog/when-not-to-use-deep-learning/


Input
Pre-Processing
Feature Mining
Feature Selection
Feature Reduction
Modelling
Accepted Results?
NO
YES
Post-Processing
Output


weights
new feature a1
weights
Bedrooms
3
new feature a2
Sq. Feet
2000
Final Price Estimate
new feature a3
Neighborhood (mapped to an id number)
1
new feature a4
Input features
Hidden layer
Output Prediction (ŷ)


Input features
Hidden layer
Output
x_1
x_2

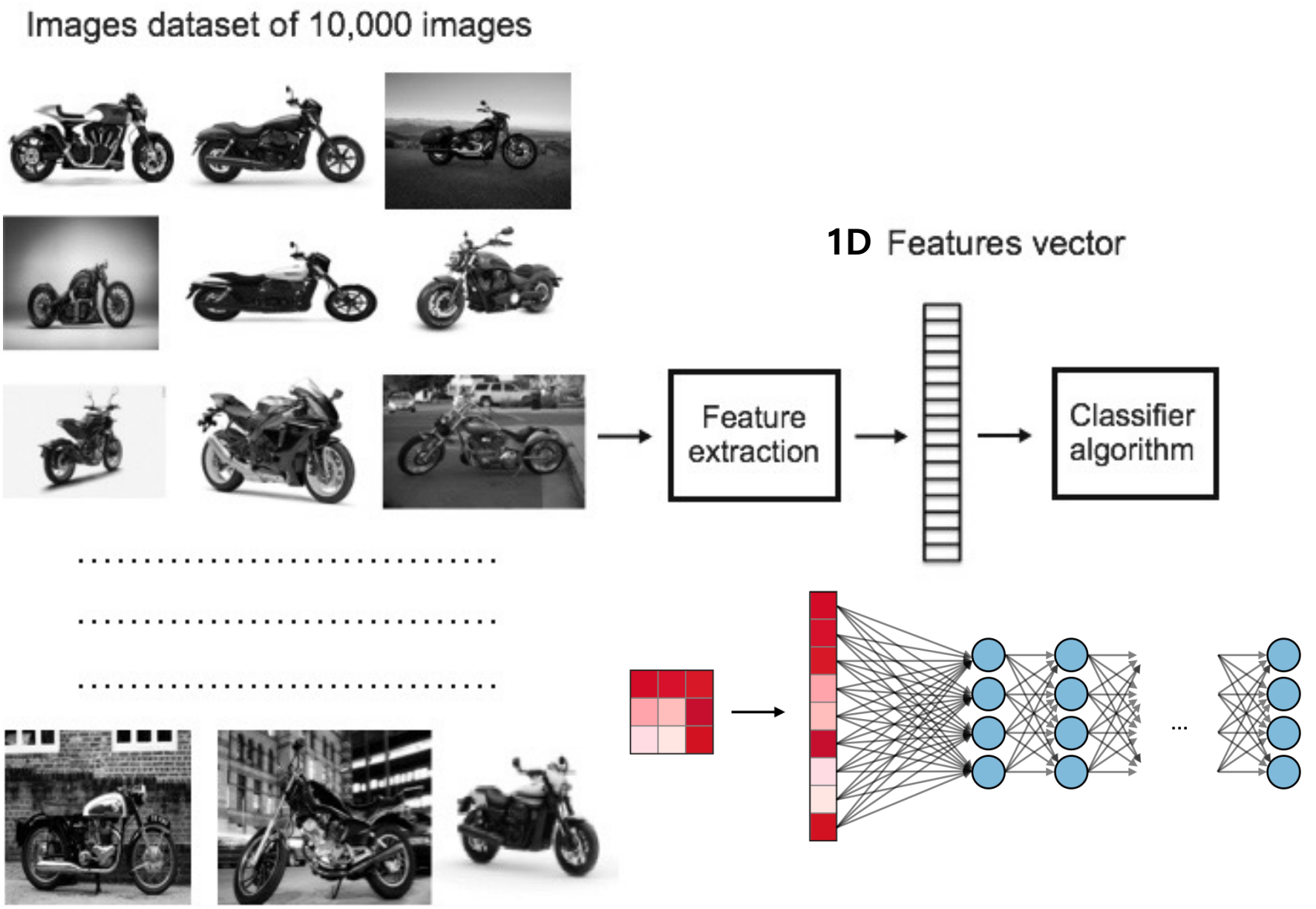
Feature Crosses

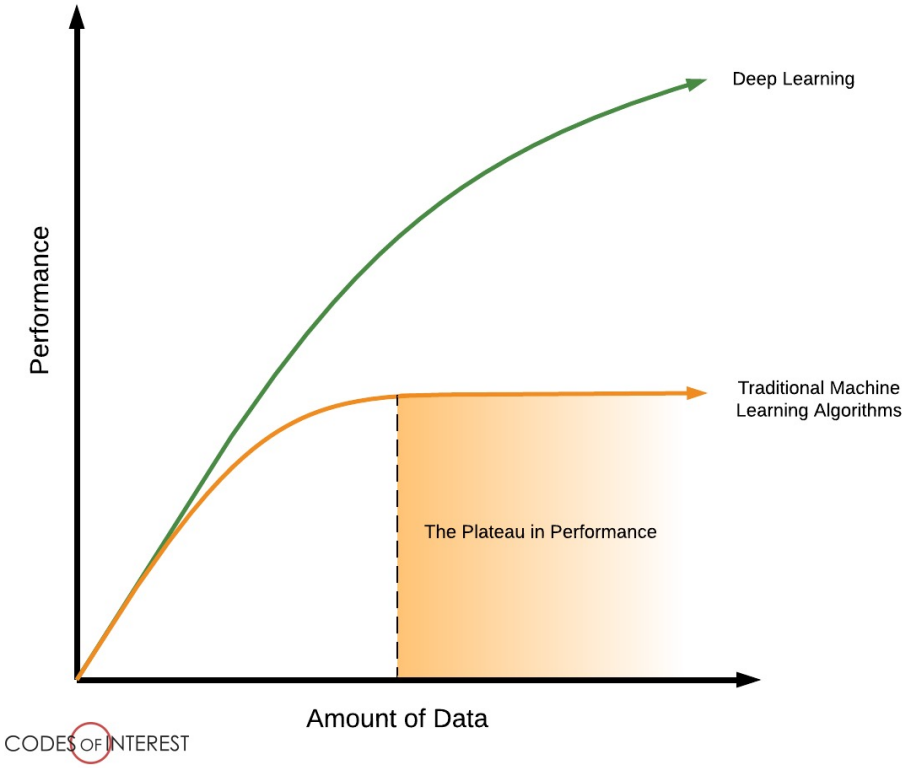
A **feature cross** is a **synthetic feature** formed by multiplying (crossing) two or more features. Crossing combinations of features can provide predictive abilities beyond what those features can provide individually.

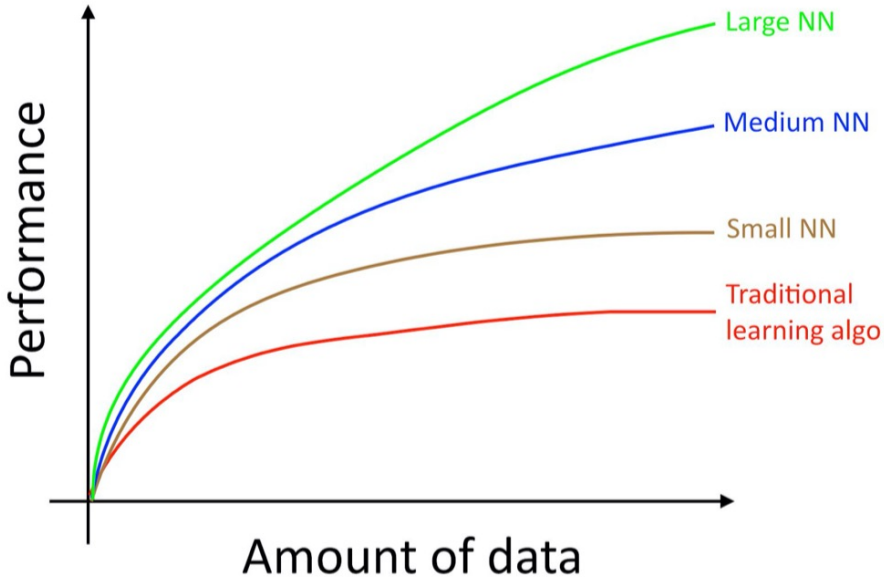
Kinds of feature crosses

We can create many different kinds of feature crosses. For example:

- [A X B]: a feature cross formed by multiplying the values of two features.
- [A x B x C x D x E]: a feature cross formed by multiplying the values of five features.
- [A x A]: a feature cross formed by squaring a single feature.

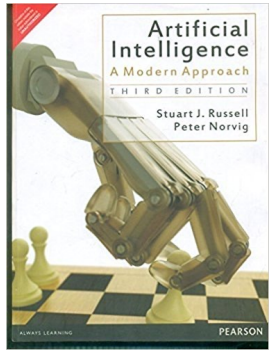

Images dataset of 10,000 images
1D Features vector
Feature extraction
Classifier algorithm


Deep Learning
Traditional Machine Learning Algorithms
The Plateau in Performance
Performance
Amount of Data
CODES OF INTEREST


Large NN
Medium NN
Small NN
Traditional learning algo
Performance
Amount of data

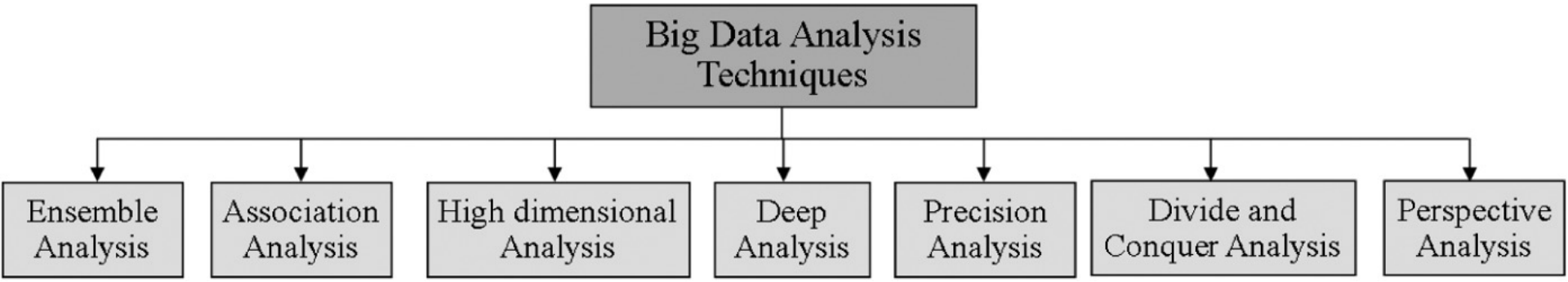
Big Data

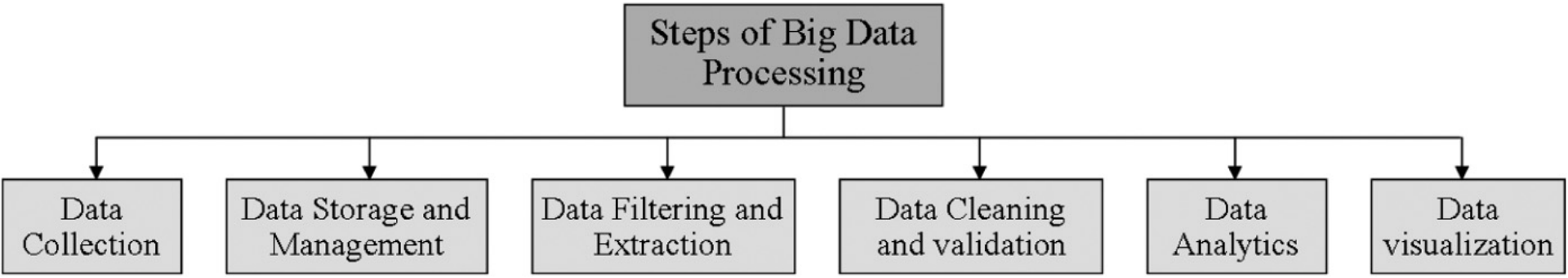
The Unreasonable Effectiveness of Data (2009)
Peter Norvig



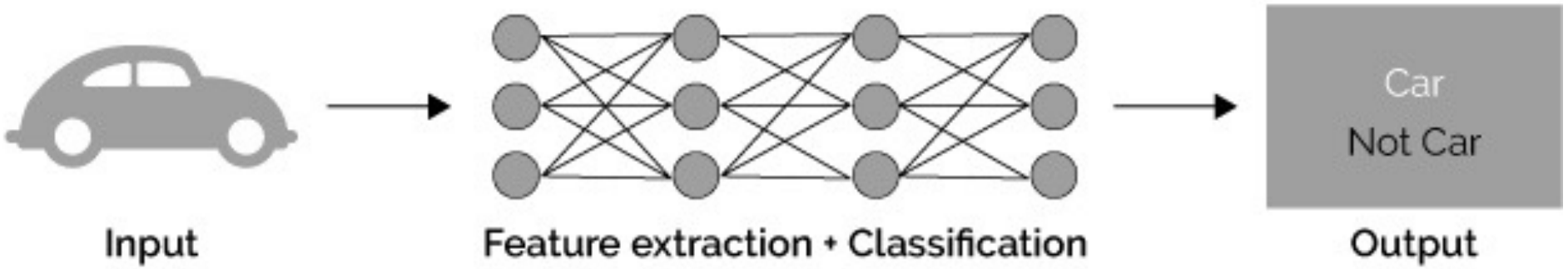
Learning Curves for Confusion Set Disambiguation (2001)
Michele Banko & Eric Brill

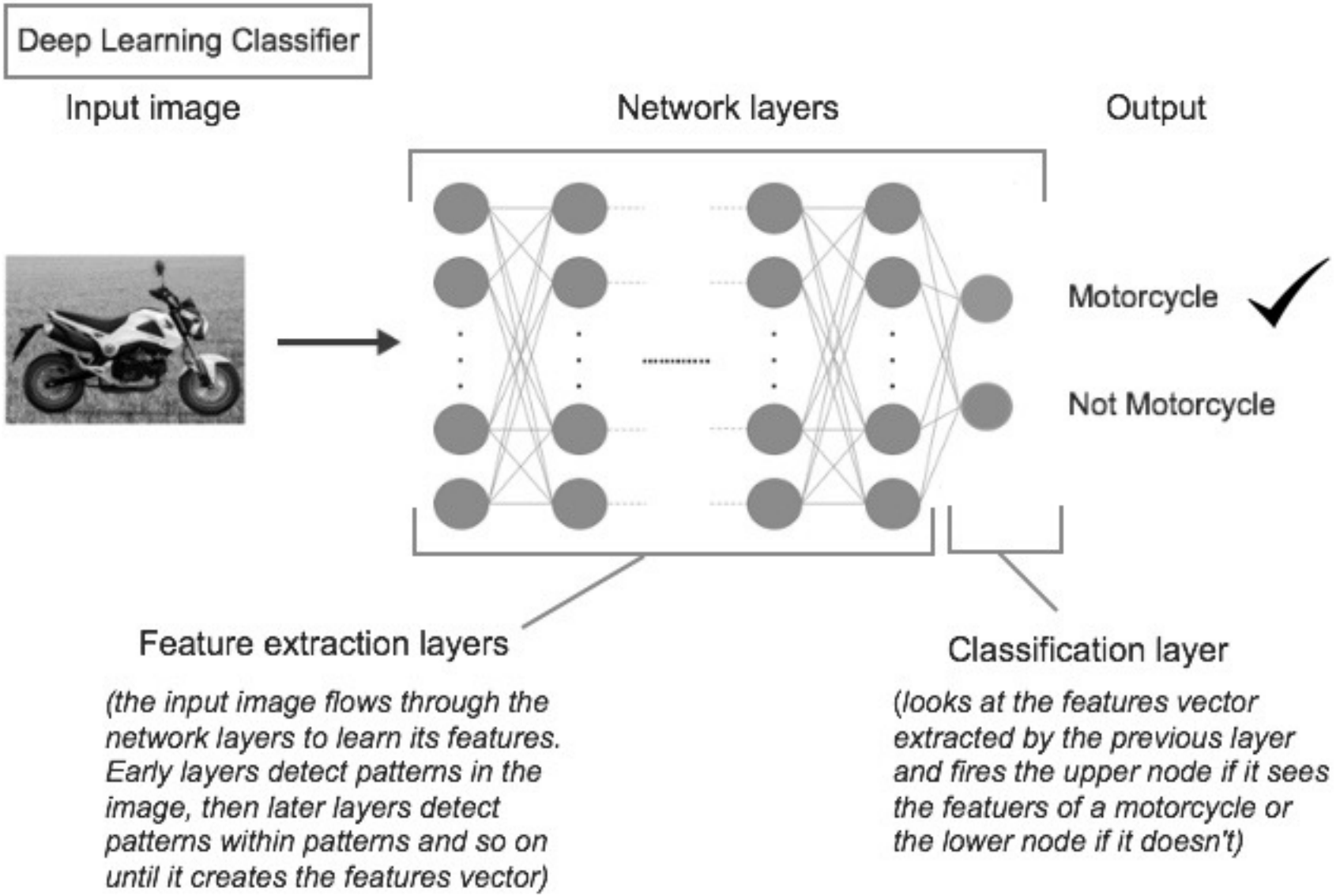
Big data	Small Data
No overfitting to training data 	Overfitting to training data is possible 
Working with complex models for difficult problems	Working with simple models for easy problems
Higher computational cost → Slower learning time 	Lower computation cost → Faster learning time 
Large-scale storage is required 	Small-scale storage is fine 


Big Data Analysis Techniques
Ensemble Analysis
Association Analysis
High dimensional Analysis
Deep Analysis
Precision Analysis
Divide and Conquer Analysis
Perspective Analysis

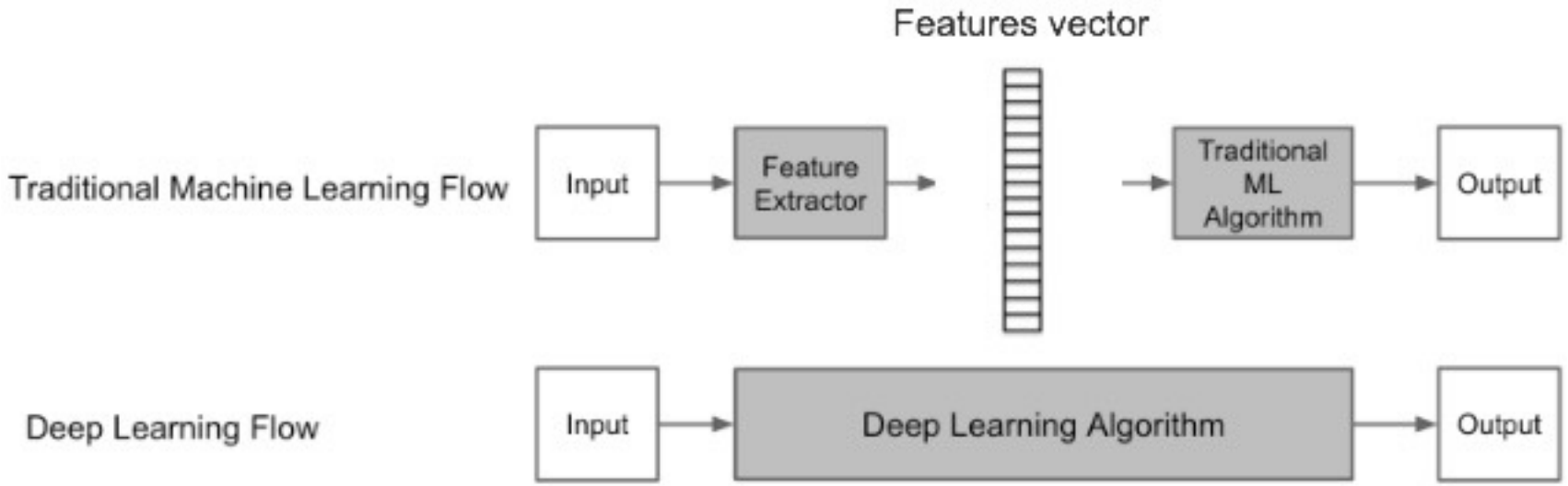

Steps of Big Data Processing
Data Collection
Data Storage and Management
Data Filtering and Extraction
Data Cleaning and validation
Data Analytics
Data visualization

- The Promise of Automatic Feature Extraction. Features can be automatically learned and extracted from raw image data.
- The Promise of End-to-End Models. Single end-to-end models can replace pipelines of specialized models.
- The Promise of Model Reuse. Learned features and even entire models can be reused across tasks.
- The Promise of Superior Performance. Techniques demonstrate better skill on challenging tasks.
- The Promise of General Method. A single general method can be used on a range of related tasks.


Car
Not Car
Input
Feature extraction + Classification
Output


Deep Learning Classifier
Input image
Network layers
Output
Motorcycle
Not Motorcycle
Feature extraction layers
(the input image flows through the network layers to learn its features. Early layers detect patterns in the image, then later layers detect patterns within patterns and so on until it creates the features vector)
Classification layer
(looks at the features vector extracted by the previous layer and fires the upper node if it sees the featuers of a motorcycle or the lower node if it doesn't)


사람이 생각한 알고리즘
결과
사람이 생각한 특징
(SIFT, HOG 등)
기계학습
(SVM, KNN 등)
결과
신경망(딥러닝)
결과


Features vector
Traditional Machine Learning Flow
Input
Feature Extractor
Traditional ML Algorithm
Output
Deep Learning Flow
Input
Deep Learning Algorithm
Output

CV Application

- Object Classification: What broad category of object is in this photograph?
- Object Identification: Which type of a given object is in this photograph?
- Object Verification: Is the object in the photograph?
- Object Detection: Where are the objects in the photograph?
- Object Landmark Detection: What are the key points for the object in the photograph?
- Object Segmentation: What pixels belong to the object in the image?
- Object Recognition: What objects are in this photograph and where are they?

classification

["people", "cars"]

People

Cars

identification

Person #1

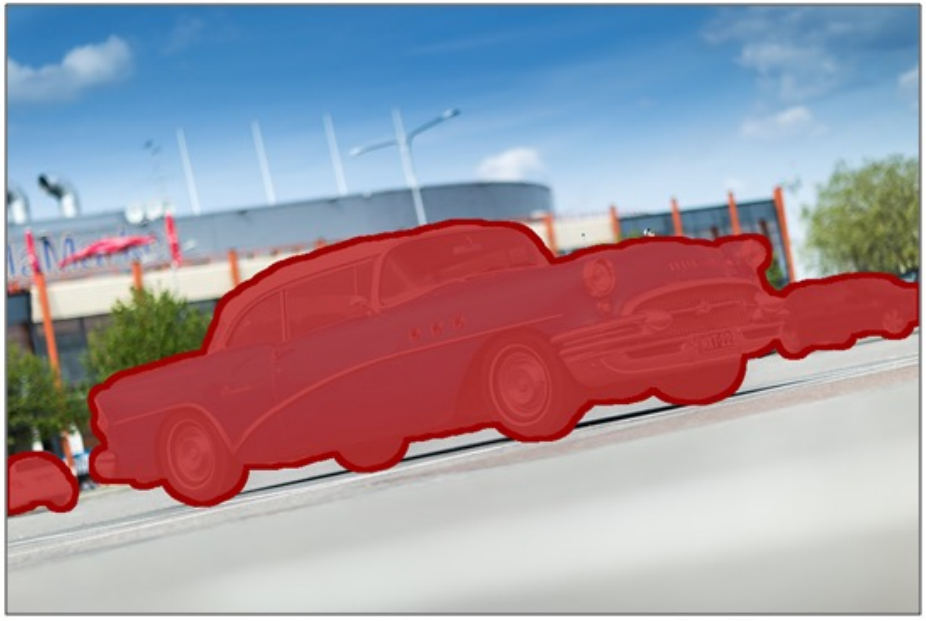
Person #2

Person #3

detection

"car"

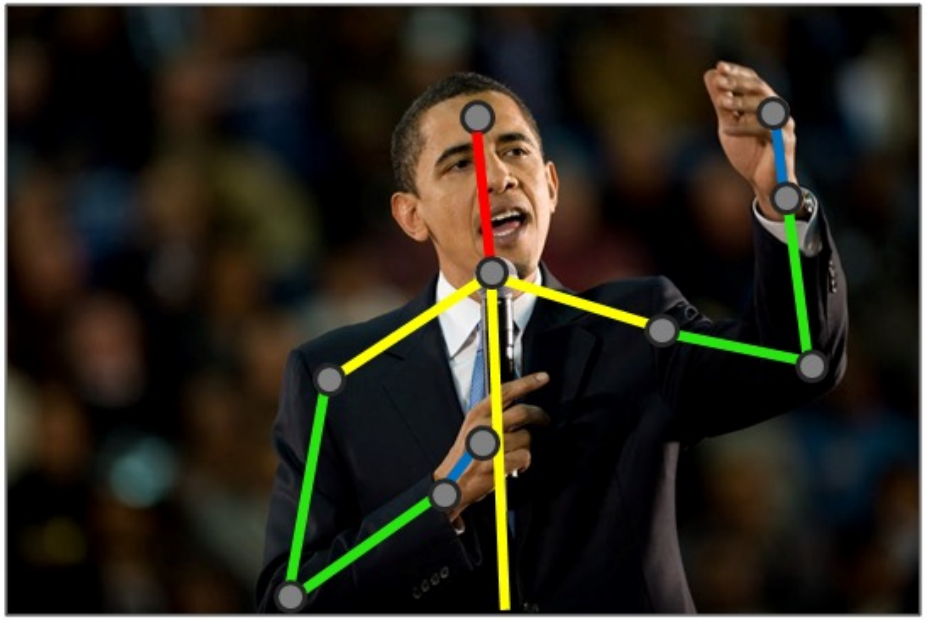
object segmentation



instance segmentation

rigid object pose estimation

human pose estimation



A person riding a motorcycle on a dirt road.

A group of young people playing a game of frisbee.

Original image

Style

Generated art

+

=

This small blue bird has a short pointy beak and brown on its wings

This bird is completely red with black wings and pointy beak