

AI Overview

목차

01

인공지능의 등장

02

인공지능의 역사

03

인공지능의 발전 단계

04

AI 반도체

05

Deep Learning

인공지능의 등장



- 1956년 다트머스대학교에서 '지능을 가진 기계'를 주제로 학술회의가 열림
- “학습의 모든 측면, 혹은 지능의 모든 특성이 원칙적으로 정확히 기술되어서 이를 모사하는 기계를 만들 수 있다는 가정에 토대를 두고 연구를 진행할 것”
- 사람의 다양한 능력을 컴퓨터가 대신할 수 있도록 하는 것
- 인공지능 (Artificial Intelligence)이라는 용어를 고안함.
- 인간의 두뇌구조를 본뜬 인공 신경망 (Artificial Neural Network)모델 등장
- 인공 신경망의 초기 모델은 퍼셉트론 (Perceptron, 1958)

관점 (Perspective)

개념 설명 (Concept Description)

사전적 개념 (Dictionary Concept)

철학적인 개념으로, 지성을 갖춘 존재 또는 시스템에 의해 만들어진 인공적인 지능을 의미

전통적 개념 (Traditional Concept)

컴퓨터가 인간의 지능적인 행동을 모방할 수 있도록 하는 소프트웨어로, 인간이 가진 지적 능력의 일부 또는 전체를 구현한 것

기술적 개념 (Technical Concept)

인간의 지능으로 할 수 있는 사고, 학습, 자기계발 등을 컴퓨터가 할 수 있도록 하는 방법을 연구하는 컴퓨터공학 및 정보기술의 한 분야

AI 개념의 직관적 이해

초지능
미지의 존재



일반 지능
인간

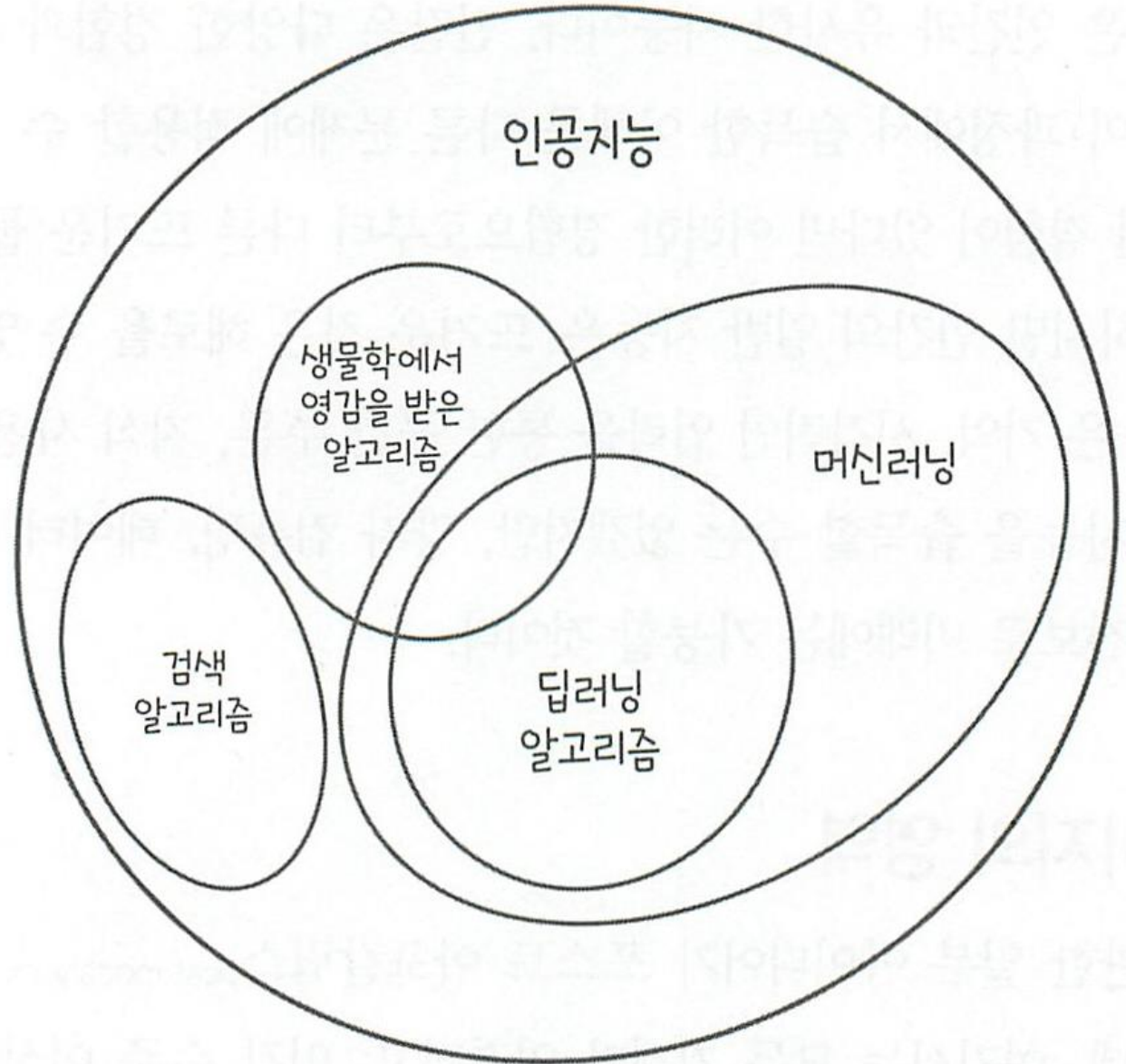


좁은 지능

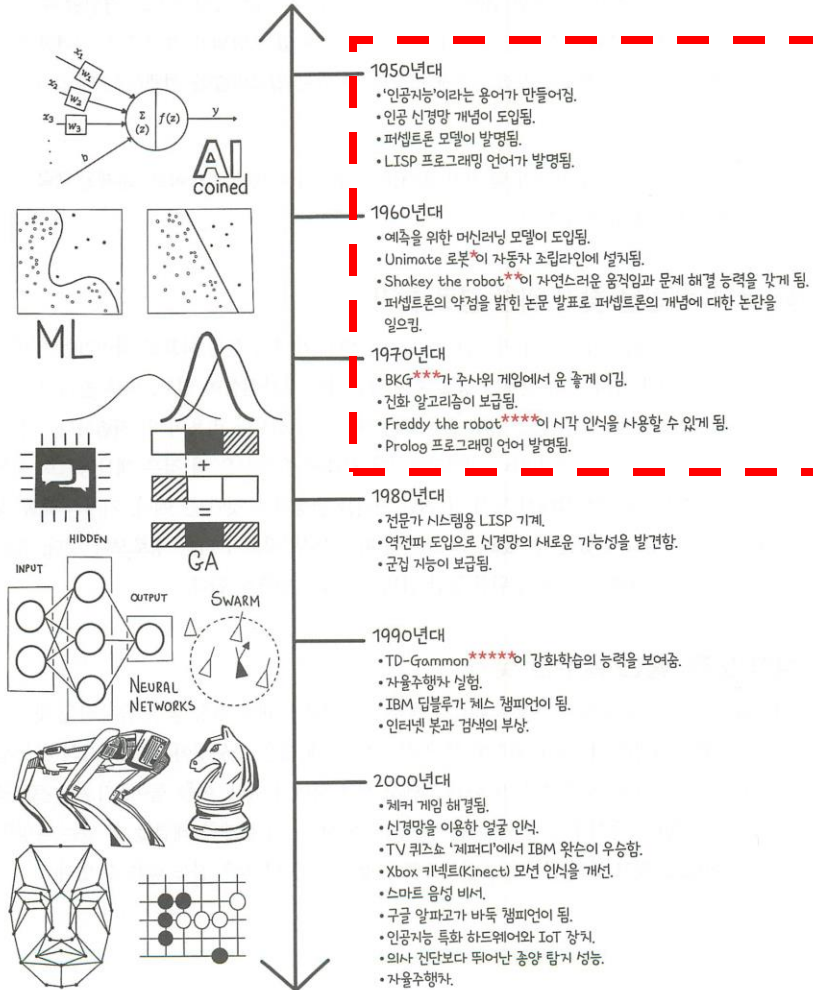
탁구 게임 프로그램
지도 경로 찾기 프로그램
금융사기 감지 프로그램



AI 개념 분류



인공지능의 역사 #1



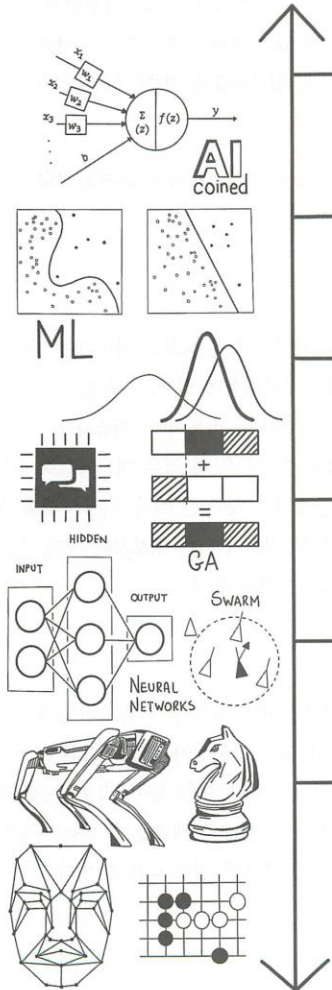
• 태동기 (1940년대 ~ 1970년대 중반):

- 초기 개념 정립: 1940년대에 앨런 튜링이 '튜링 테스트'를 제안
- '인공지능' 용어 탄생: 1956년 다트머스 회의에서 '인공지능' 용어 사용
- 초기 연구: LISP 언어 개발 및 퍼셉트론 모

• 첫 번째 'AI 겨울' (1970년대 중반 ~ 1980년대 초반):

- 초기 AI 기술의 제한으로 연구 자금이 감소하고 활동이 위축된 시기입니다. '조합 폭발(Combinatorial explosion)' 문제 등이 주요 한계로 지적되었습니다.

인공지능의 역사 #2



• 전문가 시스템의 부상과 신경망 연구 재개 (1980년대):

• **전문가 시스템:** 특정 분야의 전문 지식을 활용한 '전문가 시스템'들이 상업적으로 성공하면서 AI에 대한 관심이 재점화 되었습니다. (예: MYCIN, PROSPECTOR)

• **신경망 모델 발전:** 다층 퍼셉트론(MLP)과 오차 역전파 알고리즘의 발전으로 신경망 연구가 활기를 되찾았습니다

• 두 번째 'AI 겨울' (1980년대 후반 ~ 1990년대 중반):

• 전문가 시스템의 한계와 새로운 기술 부족으로 AI 연구가 재차 침체되었습니다.

1950년대

- '인공지능'이라는 용어가 만들어짐.
- 인공 신경망 개념이 도입됨.
- 퍼셉트론 모델이 발명됨.
- LISP 프로그래밍 언어가 발명됨.

1960년대

- 예측을 위한 머신러닝 모델이 도입됨.
- Unimate 로봇*이 자동차 조립라인에 설치됨.
- Shakey the robot*이 자연스러운 움직임과 문제 해결 능력을 갖게 됨.
- 퍼셉트론의 약점을 밝힌 논문 발표로 퍼셉트론의 개념에 대한 논란을 일으킴.

1970년대

- BKG***가 추사위 게임에서 윤 출제 이김.
- 진화 알고리즘이 보급됨.
- Freddy the robot***이 시각 인식을 사용할 수 있게 됨.
- Prolog 프로그래밍 언어 발명됨.

1980년대

- 전문가 시스템용 LISP 기계.
- 역전파 도입으로 신경망의 새로운 가능성을 발견함.
- 군집 지능이 보급됨.

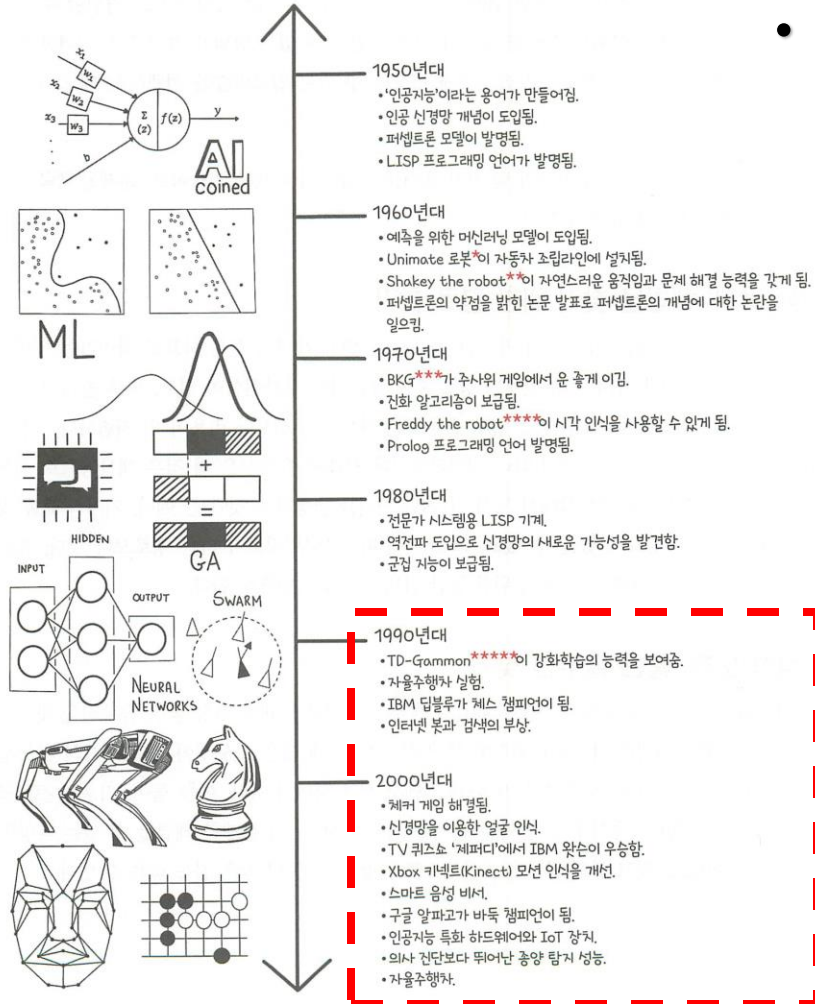
1990년대

- TD-Gammon****이 강화학습의 능력을 보여줌.
- 자율주행차 실험.
- IBM 딥블루가 체스 챔피언이 됨.
- 인터넷 붐과 검색의 부상.

2000년대

- 체커 게임 해결됨.
- 신경망을 이용한 얼굴 인식.
- TV 퀴즈쇼 '제퍼디!'에서 IBM 왓슨이 우승함.
- Xbox 키넥트(Kinect) 모션 인식을 개편.
- 스마트 음성 비서.
- 구글 알파고가 바둑 챔피언이 됨.
- 인공지능 특화 하드웨어와 IoT 장치.
- 의사 진단보다 뛰어난 중앙 탐지 성능.
- 자율주행차.

인공지능의 역사 #3



• 조용한 발전과 딥러닝의 시작 (1990년대 중반 ~ 2010년대 초반):

- 데이터와 머신러닝의 중요성이 강조되며, 인터넷의 확산과 데이터 양의 급증으로 머신러닝 기술이 주목받기 시작했습니다.
- 딥러닝의 기초가 되는 기술로 심층 오토인코더와 심층 신뢰 신경망(DBN)이 등장했습니다.

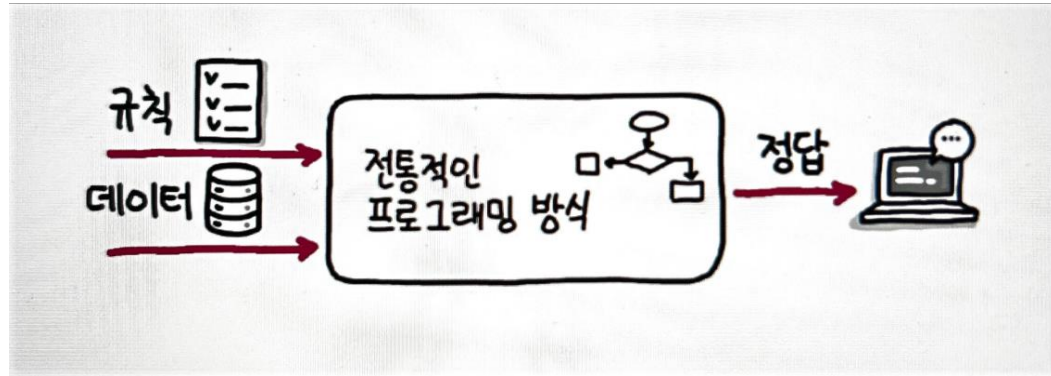
인공지능의 역사 #4

- 딥러닝의 부활과 AI의 대중화 (2012년 이후):
 - GPU 발전: GPU의 발전이 딥러닝 알고리즘의 병렬 처리를 가능하게 했습니다.
 - 빅데이터: 인터넷과 모바일 기기의 확산으로 대량의 데이터가 축적되어 딥러닝 모델 학습에 활용됩니다.
 - 알고리즘 발전: ImageNet에서의 성과를 통해 딥러닝 알고리즘이 크게 발전했습니다.
 - 알파고의 등장: 2016년 구글의 알파고가 이세돌과의 대결에서 승리하며 AI 연구에 대한 관심과 투자가 증가했습니다.
 - 생성형 AI의 시대: 최근 GPT와 같은 대규모 언어 모델이 등장하여 다양한 창작 활동이 가능해지고 있습니다.

인공지능 발전 단계

- 약인공지능 (Narrow AI / Weak AI)
 - 특정 작업에 특화된 AI로, 음성 비서나 이미지 인식, 번역 프로그램 등을 포함합니다.
- 강인공지능 (General AI / Strong AI / AGI)
 - 인간처럼 다양한 지적 활동을 수행하고 스스로 학습하는 AI로, 아직 개발되지 않은 기술입니다.
- 초인공지능 (Super Intelligence / ASI)
 - 인간의 지능을 모든 면에서 초월하는 AI로, 주로 SF 영화에서 다루어집니다.

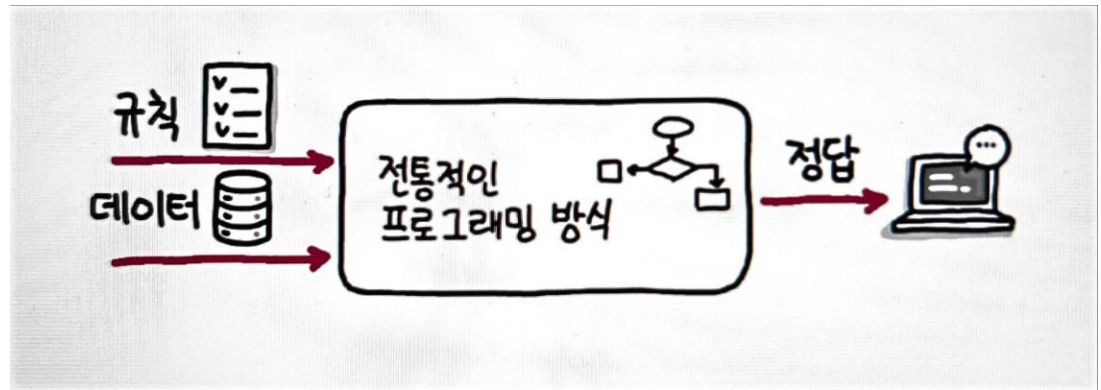
규칙 기반, 인공지능의 구현



- 세계 최초의 프로그래머로 알려진 **에이다 러브 레이스**(Ada Lovelace, 1815~52)는 컴퓨터가 개발되기 이전에 프로그래밍의 기본 원리를 확립했습니다.
- 그녀는 19세기 당시 "**기계가 앞으로 정교하게 작곡할 수 있을 것**"이라고 예측했습니다.
- 하지만 그녀는 "**기계는 인간이 지시한 작업만 수행하며, 어떠한 해석이나 진실을 예측할 능력은 없다**"고 언급했습니다.

규칙 기반, 인공지능의 구현

- 프로그래밍이란?
- 규칙(알고리즘) + 데이터(자료구조)

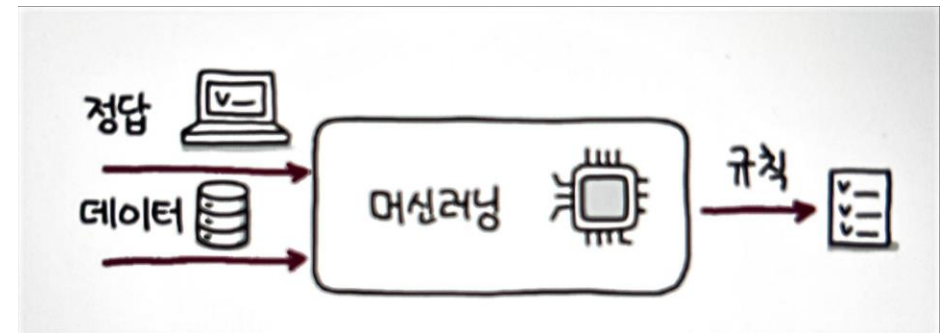
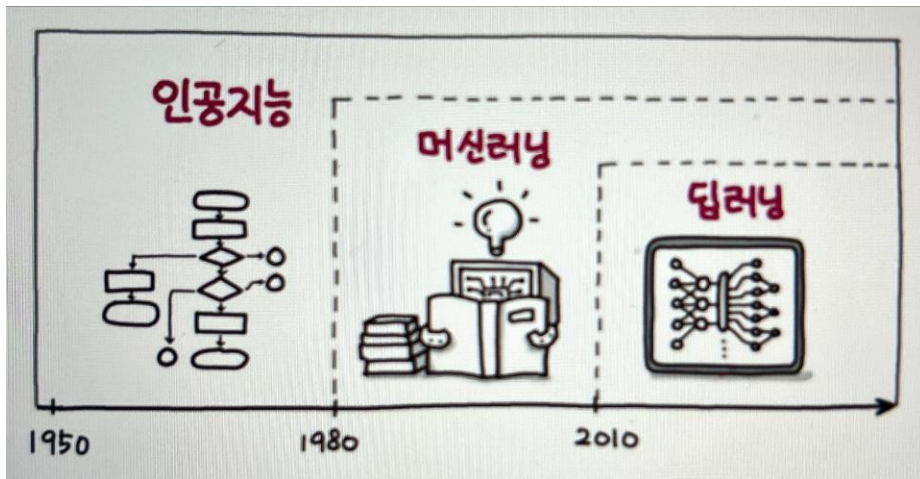


If 신호등이 (color = ??)이라면,
Then 멈춰라.

If 신호등이 (color = ??)이라면,
Then 길을 건너라.

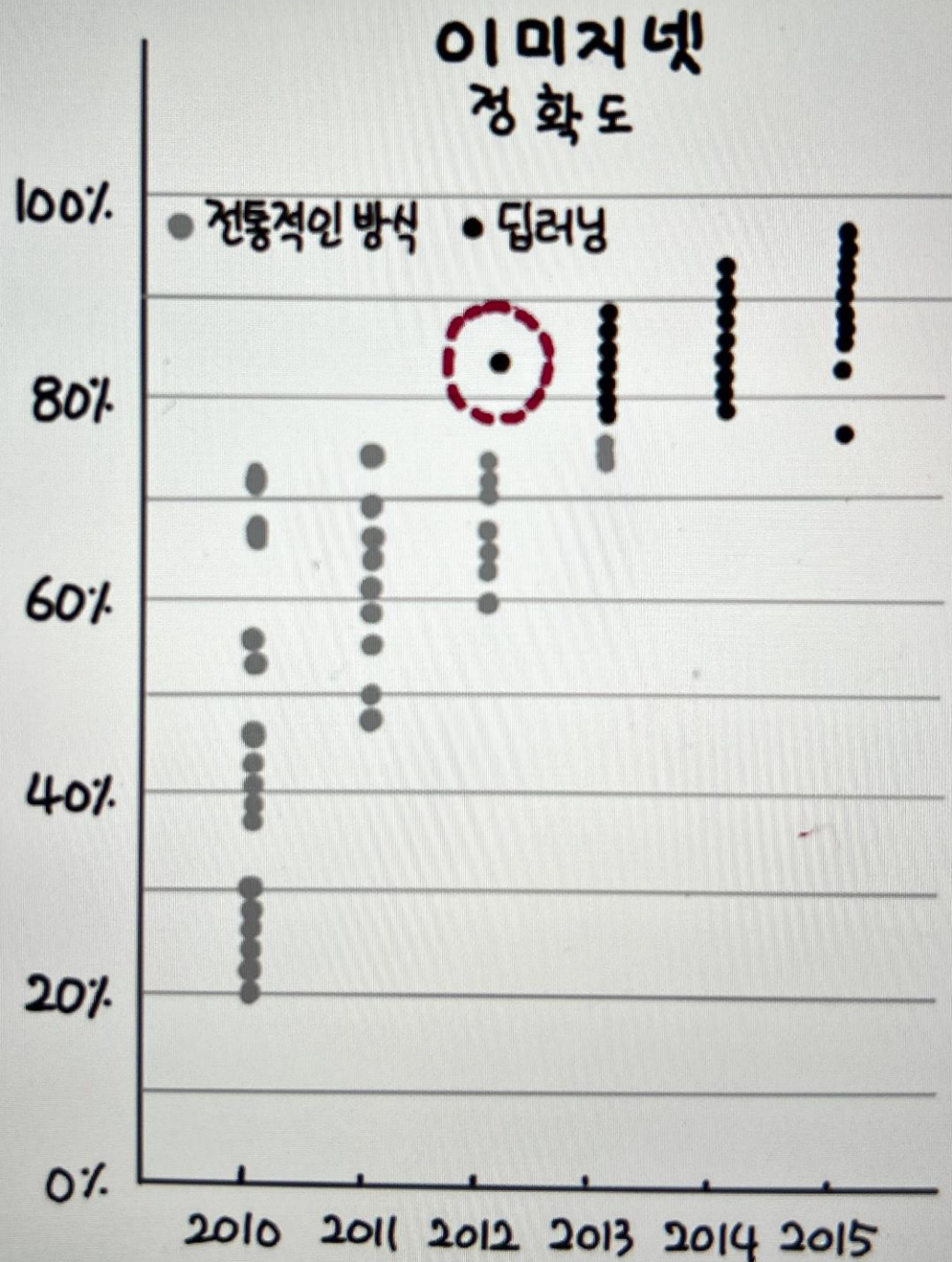
스스로 규칙을 찾다

- 1980년대 머신러닝 (Machine Learning; 기계학습)이라고 부르는 알고리즘을 활용하기 시작함.



인공신경망의 성과

- 규칙을 찾아내는 머신러닝의 시대가 열리면서 인공신경망 연구가 활기를 찾게 됨.
- 2010년, 스탠퍼드대학교의 페이페이 리(Fei-Fei Li / 1976 ~) 교수는 100만장의 이미지를 1,000개의 카테고리로 분류하는 이미지넷 대규모 시각 인식 챌린지(ILSVRC)를 주최하게 됨.
- 정답 데이터의 생성은 아마존 메케니컬 터크(Amazon Mechanical Turk)에서 담당

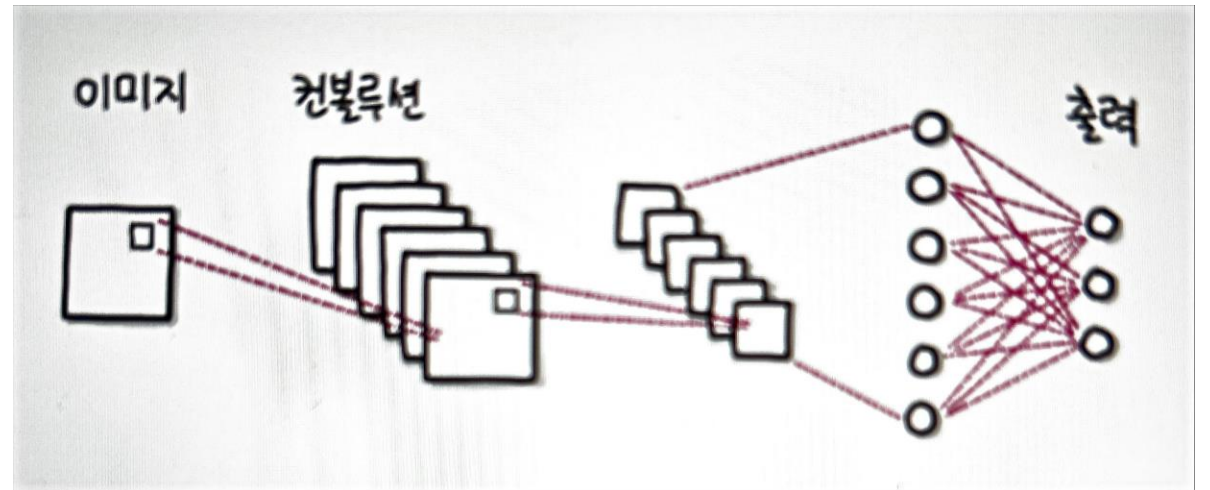


ImageNet 대회

- 대회의 정확도는 매년 1~2%씩 상승했음 (72% -> 74%).
- 2012년, 토론토대학교의 제프리 힌튼(Geoffrey Hinton)교수팀이 무려 84.7%의 정확도를 보이며 우승을 차지함.
- 딥러닝(Deep Learning) 방식으로 높은 성능 달성

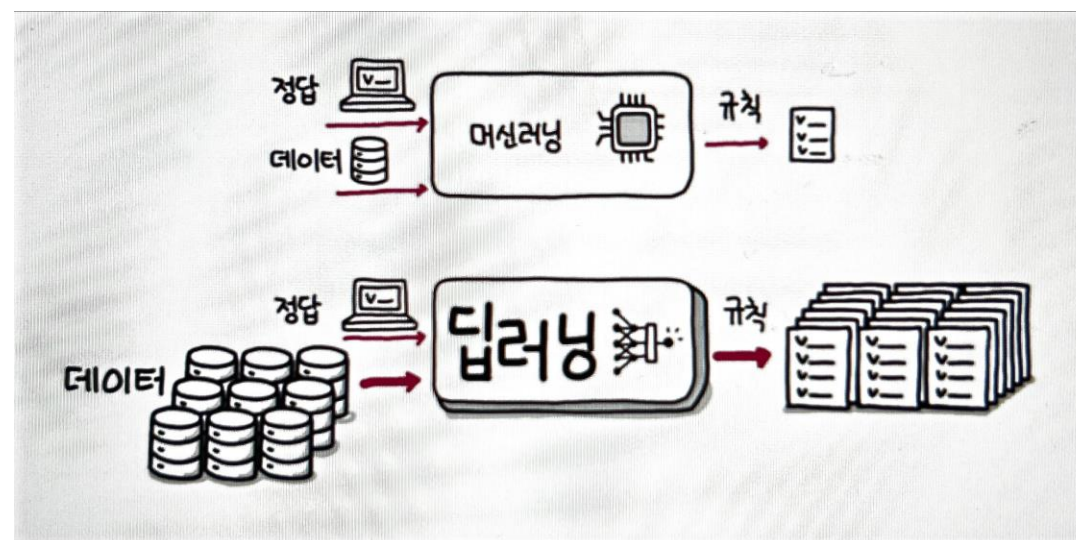
Deep Learning

- 힌튼 교수팀이 사용한 방법은 그림과 같은 컨볼루션(Convolution) 기법을 사용한 딥러닝.
- 모델의 이름은 알렉스넷(AlexNet)
- 컨볼루션 신경망은 얀 르쿤(Yann LeCun)이 정립



Deep Learning & Data

- 알고리즘의 발전과 제대로 학습할 수 있는 방법을 찾아내자 딥러닝이라는 새로운 이름을 부여받게 됨.
- 사람의 두뇌를 흉내낸 방식
 - GPT-3는 매개변수 1,750억
 - 사람은 약 1,000억개의 뉴런

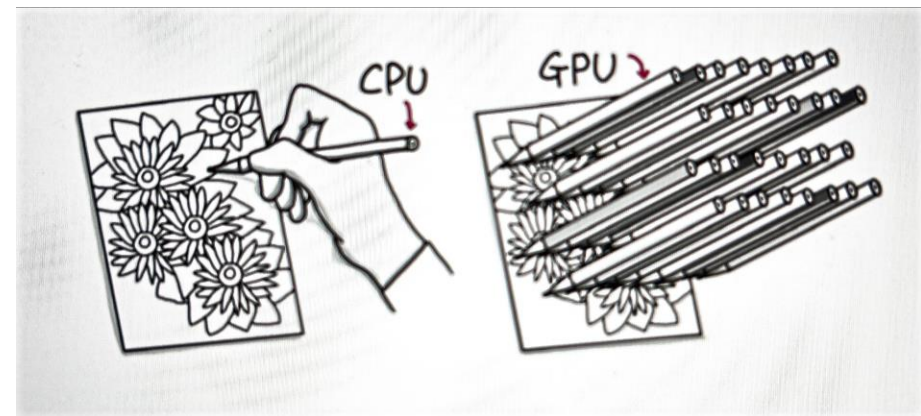
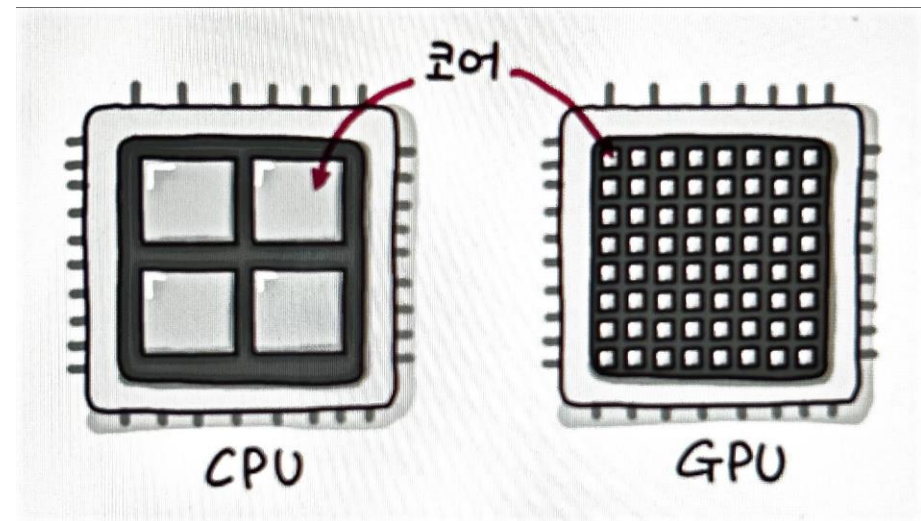


GPU가 완성한 AI

게임과 그래픽 카드가 인공지능과
무슨 관련이 있을까?



GPU vs CPU



GPU, 왜 AI에 필수적인가?

- AI, 특히 딥러닝 모델은 **병렬 연산**에 최적화되어 있으며, GPU는 수천 개의 코어로 많은 연산을 동시에 처리합니다. 이 **대규모 병렬 처리 능력** 덕분에 GPU는 AI 학습과 추론에 필수적입니다.
 - **딥러닝 학습 가속화:** 복잡한 모델 학습에 필요한 행렬 곱셈을 GPU가 빠르게 처리하여 학습 시간을 크게 단축시킵니다.
 - **실시간 AI 서비스 가능:** 자율주행차와 같은 AI 서비스에서 GPU는 실시간 데이터 처리를 지원하여 빠른 판단을 가능하게 합니다.
 - **AI 생태계 확장:** NVIDIA와 같은 GPU 제조사는 CUDA와 같은 소프트웨어 플랫폼을 제공해 AI 개발자들이 GPU를 효율적으로 활용하도록 돕고 있습니다.

Home Work

폰 노이만 아키텍처

폰 노이만 병목 현상

캐시 메모리

하버드 아키텍처

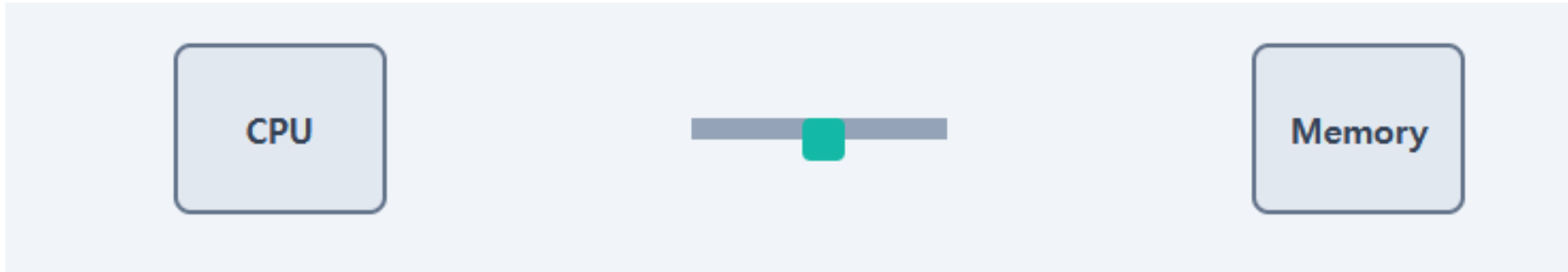
파이프라이닝

멀티코어 프로세서

왜 AI 반도체인가?

- AI 연산은 기존의 범용 프로세서(CPU)로는 감당하기 힘든 독특한 특징을 가집니다.
- 전통적인 컴퓨터 구조는 CPU와 메모리 사이의 데이터 이동 경로가 하나뿐이라, 대규모 데이터를 처리해야 하는 AI 연산 시 심각한 성능 저하가 발생합니다. 이를 '폰 노이만 병목 현상'이라고 합니다.

왜 AI 반도체인가?



- CPU가 연산을 하려면 메모리에서 데이터를 가져와야 하고, 결과를 다시 메모리에 저장해야 합니다.
- AI는 이 과정이 수없이 반복되어 데이터 이동 경로에 병목이 생깁니다.

반도체 유형

범주	CPU (중앙 처리 장치)	GPU (그래픽 처리 장치)	TPU (텐서 처리 장치)	ASIC (주문형 반도체)	FPGA (필드 프로그래머블 게이트 어레이)	NPU (신경망 처리 장치)
핵심 특징	범용, 순차 처리, 소수 빅코어	병렬 처리, 수천 개 리틀코어	텐서 연산 특화, 구글 맞춤형	특정 작업 맞춤 설계	하드웨어 재구성 가능	신경망 연산 특화, 뇌 모방
장점	높은 유연성, 다양한 작업 처리	뛰어난 병렬 처리, AI 학습 및 추론 고성능	고속 행렬 연산, 높은 AI 연산 성능	최고 성능/효율성 (목표 작업)	하드웨어 재구성 가능, 유연성 높음.	높은 전력 효율성, 실시간 추론
단점	AI 연산에 낮은 에너지 효율성, 낮은 계산 성능	높은 전력 소모, 발열 심함	높은 비용(클라우드 서비스 유리)	높은 개발 비용, 긴 개발 기간, 설계 변경 어려움	복잡한 로직 설계, 칩당 단가 높음.	활용 폭 좁음 (AI 연산에만 특화).
주요 활용 분야	일반 컴퓨팅, 복잡한 제어	딥러닝 학습, 대규모 추론, 자율주행, 클라우드 AI	구글 AI 모델 학습, 데이터 센터 연산 최적화	임베디드 시스템, 대량 생산 제품	고주파수 트레이딩, 실시간 의료 영상 분석	모바일 기기, IoT, 엣지 컴퓨팅
아키텍처 초점	범용	병렬 연산	텐서 연산	맞춤형	재구성 가능	신경망

Deep Learning Framework



주요 프레임워크 비교 #1

TensorFlow

PyTorch

Keras

TensorFlow

Google에서 개발한 딥러닝 프레임워크로, 강력한 확장성과 생산 환경 배포를 위한 풍부한 생태계가 특징입니다. TFX, TensorFlow.js, TensorFlow Lite 등 다양한 플랫폼을 지원하여 아이디어 구상부터 대규모 서비스까지 전 과정을 커버합니다.

주요 사용 사례

Google

검색, 번역, 포토 등 핵심 서비스 전반

Airbnb

이미지 분류 및 객체 탐지를 통한 서비스 개선

Twitter

사용자 타임라인의 트윗 랭킹 시스템

Coca-Cola

모바일 앱의 구매 증명 이미지 인식

장점

- ✓ 강력한 MLOps 및 배포 도구(TFX, Serving)
- ✓ 모바일/웹/엣지 등 다양한 플랫폼 지원
- ✓ 대규모 분산 학습 및 TPU 하드웨어 최적화
- ✓ 가장 큰 커뮤니티와 풍부한 레퍼런스

단점

- ✗ 초기 버전의 가파른 학습 곡선 (현재 많이 개선됨)
- ✗ 상대적으로 무겁고 장황한 코드 스타일
- ✗ 유연성 측면에서 PyTorch보다 다소 경직됨

주요 프레임워크 비교 #2

TensorFlow

PyTorch

Keras

PyTorch

Meta(구 Facebook)에서 개발했으며, 파이썬 친화적이고 직관적인 API로 연구자들 사이에서 큰 인기를 얻고 있습니다. Define-by-Run 방식을 채택하여 모델을 유연하게 구축하고 디버깅하기 용이합니다.

주요 사용 사례

Meta

Facebook, Instagram 등 서비스의 AI 기능

Tesla

자율 주행 시스템의 컴퓨터 비전 모델 개발

Microsoft

Azure 클라우드 플랫폼에서 PyTorch 적극 지원

OpenAI

GPT와 같은 LLM 초기 연구 및 개발

장점

- ✓ 파이썬다운 직관적이고 간결한 코드
- ✓ 동적 계산 그래프로 높은 유연성 및 쉬운 디버깅
- ✓ 학계 및 연구 커뮤니티의 압도적인 지지
- ✓ Hugging Face 등 최신 AI 라이브러리와의 뛰어난 호환성

단점

- ✗ TensorFlow 대비 배포 및 운영 생태계가 상대적으로 미흡
- ✗ 과거 시각화 도구(TensorBoard) 지원 부족 (현재 통합됨)
- ✗ 상대적으로 적은 상업적 이용 사례 및 레퍼런스

주요 프레임워크 비교 #3

TensorFlow

PyTorch

Keras

Keras

딥러닝을 더 쉽게 접근할 수 있도록 만든 고수준 API입니다. TensorFlow 위에서 동작하며(현재는 멀티 백엔드 지원), 마치 레고 블록을 조립하듯 간단한 코드로 신경망을 구축할 수 있어 입문자에게 가장 적합합니다.

주요 사용 사례

Netflix

추천 시스템의 빠른 프로토타이핑

Uber

다양한 머신러닝 문제 해결에 활용

NASA

위성 이미지 분석 및 천문학 연구

Startups

빠른 PoC(개념 증명) 모델 개발

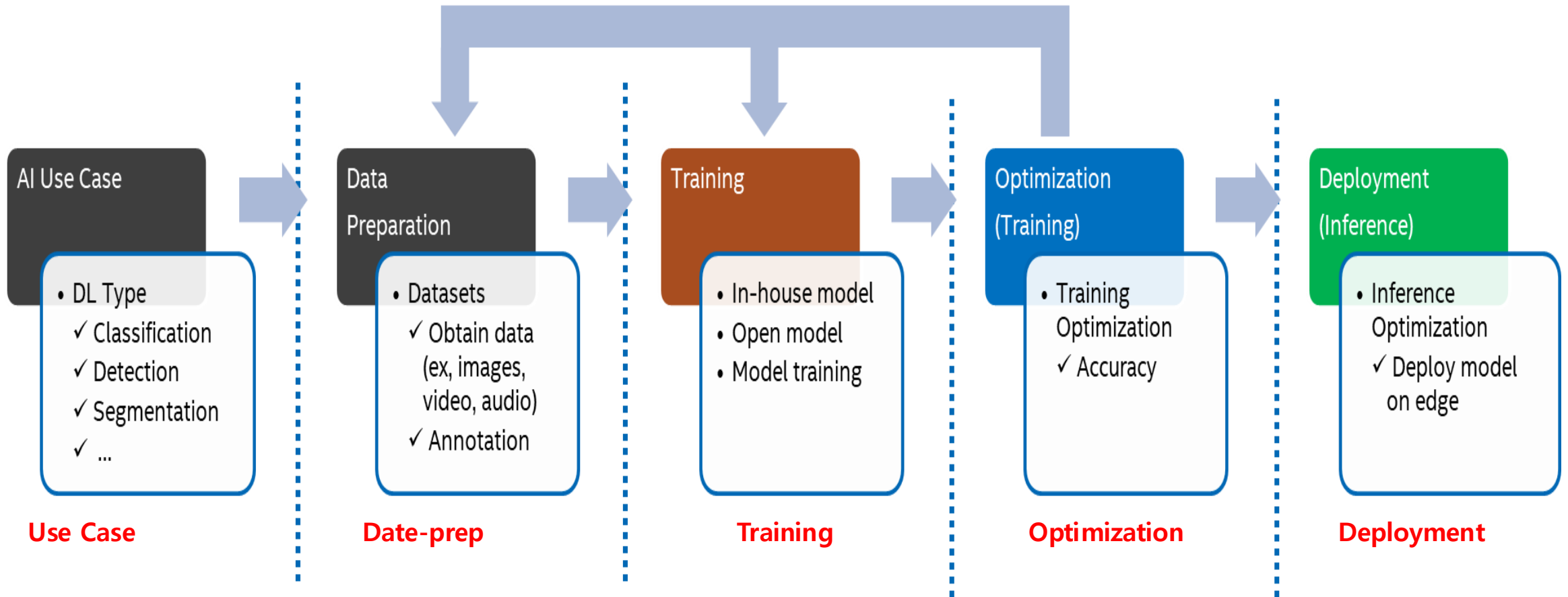
장점

- ✓ 가장 쉽고 직관적인 API로 빠른 프로토타이핑 가능
- ✓ 코드 가독성이 높고 문서화가 잘 되어 있음
- ✓ TensorFlow 2.0부터 공식 API로 통합되어 안정성 확보
- ✓ 멀티 백엔드(TF, PyTorch, JAX) 지원으로 유연성 증대

단점

- ✗ 저수준의 세밀한 모델 제어가 어려움
- ✗ 고수준 API이므로 내부 동작 이해가 어려울 수 있음
- ✗ 단독으로 사용하기보다 백엔드 프레임워크에 의존적

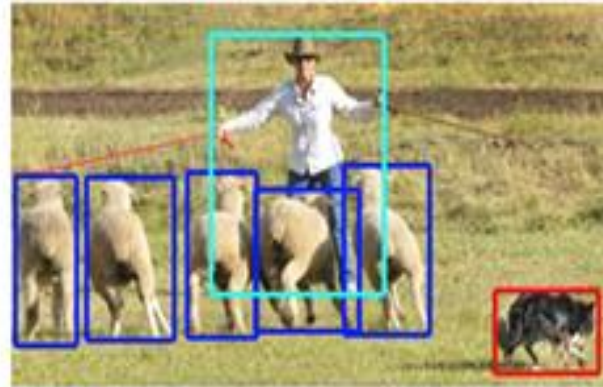
Deep Learning Workflow



1) Problem Definition



CLASSIFICATION



DETECTION



SEGMENTATION



NEURAL STYLE TRANSFER



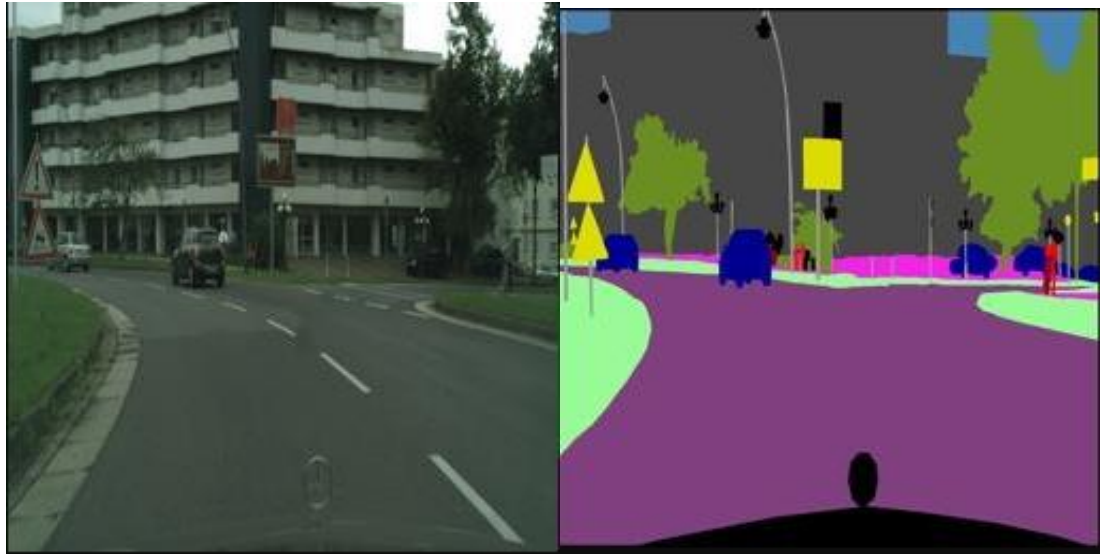
IMAGE CAPTION



COLORIZATION

2) Dataset preparation/annotation

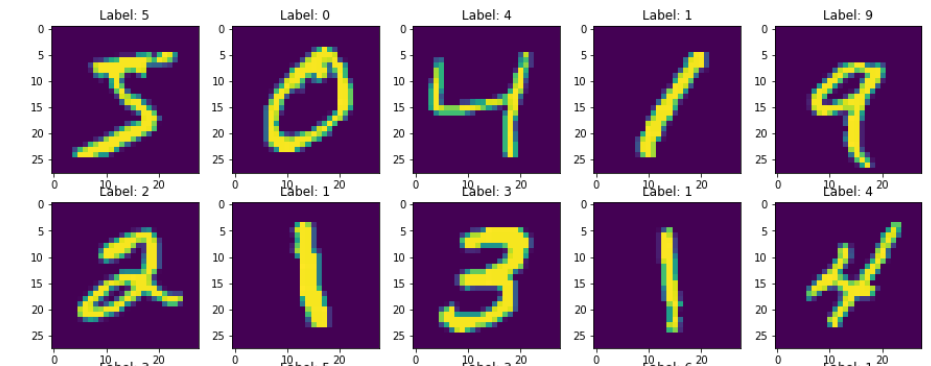
- Cityscape/Camvid
Semantic Segmentation



- MNIST

Hand-written digit

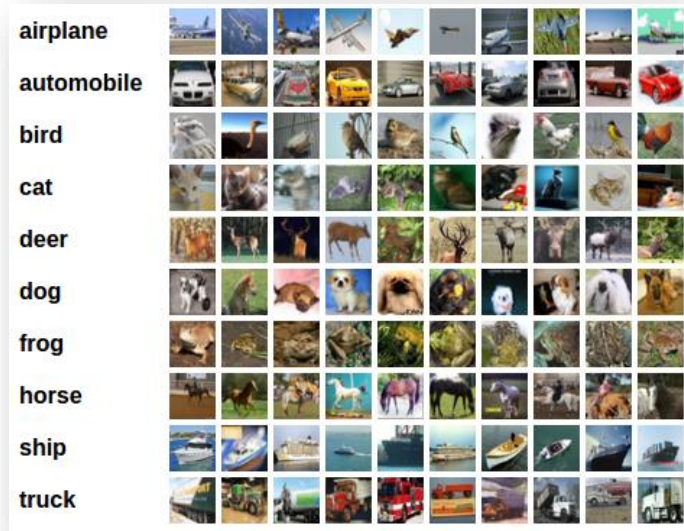
1	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]	
2	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]	
3	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]	
4	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]	
5	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]	
6	[0	0	0	0	0	0	0	0	0	0	0	3	18	18	126	136	175	26	166	255	247	127	0	0]
7	[0	0	0	0	0	0	0	30	36	94	154	170	253	253	253	253	225	172	253	242	195	64	0	0]
8	[0	0	0	0	0	0	49	238	253	253	253	253	253	253	253	253	93	82	82	56	39	0	0	0]
9	[0	0	0	0	0	0	18	219	253	253	253	253	253	198	182	247	241	0	0	0	0	0	0	0]
10	[0	0	0	0	0	0	80	156	107	253	253	205	11	0	43	154	0	0	0	0	0	0	0	0]
11	[0	0	0	0	0	0	14	1	154	253	90	0	0	0	0	0	0	0	0	0	0	0	0	0]
12	[0	0	0	0	0	0	0	0	139	253	190	2	0	0	0	0	0	0	0	0	0	0	0	0]
13	[0	0	0	0	0	0	0	0	11	190	253	70	0	0	0	0	0	0	0	0	0	0	0	0]
14	[0	0	0	0	0	0	0	0	0	0	35	241	225	160	108	1	0	0	0	0	0	0	0	0]
15	[0	0	0	0	0	0	0	0	0	0	81	240	253	253	119	25	0	0	0	0	0	0	0	0]
16	[0	0	0	0	0	0	0	0	0	0	0	45	186	253	253	150	27	0	0	0	0	0	0	0]
17	[0	0	0	0	0	0	0	0	0	0	0	16	93	252	253	187	0	0	0	0	0	0	0	0]
18	[0	0	0	0	0	0	0	0	0	0	0	0	0	249	253	249	64	0	0	0	0	0	0	0]
19	[0	0	0	0	0	0	0	0	0	0	0	46	130	183	253	253	207	2	0	0	0	0	0	0]
20	[0	0	0	0	0	0	0	0	0	39	148	229	253	253	253	250	182	0	0	0	0	0	0	0]
21	[0	0	0	0	0	0	0	24	114	221	253	253	253	253	201	78	0	0	0	0	0	0	0	0]
22	[0	0	0	0	0	0	23	66	213	253	253	253	253	198	81	2	0	0	0	0	0	0	0	0]
23	[0	0	0	0	0	18	171	219	253	253	253	253	195	80	9	0	0	0	0	0	0	0	0	0]
24	[0	0	0	55	172	226	253	253	253	253	253	244	133	11	0	0	0	0	0	0	0	0	0	0]
25	[0	0	0	136	253	253	253	212	135	132	16	0	0	0	0	0	0	0	0	0	0	0	0	0]
26	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]
27	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]
28	[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]



2) Dataset preparation/annotation – cont'd

- CIFAR-10 dataset

- Label : 10 classes
- Consists of 60,000 (32x32) color images, with 6000 images per class.

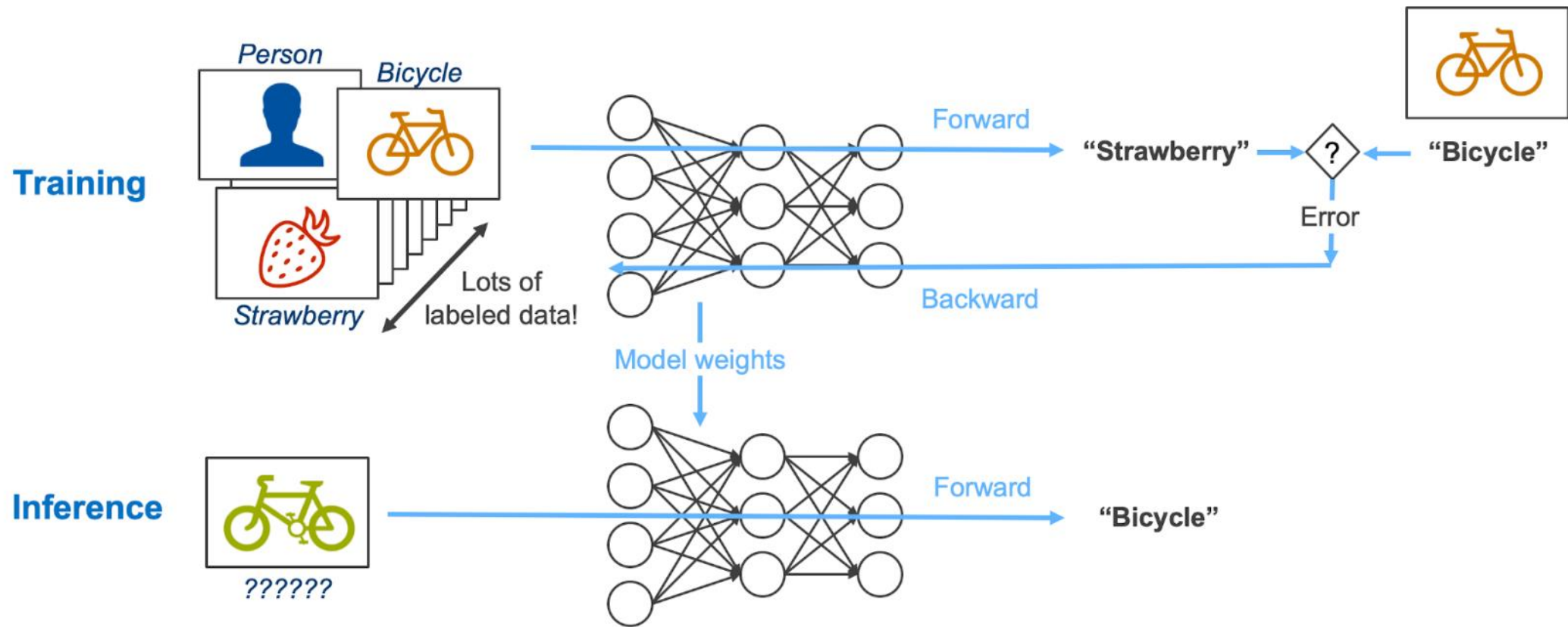


- IMAGENET

- Label : 1000 classes
- Consists of 800,000 (256x256 or 224x224) color Images, with 800 images per class.



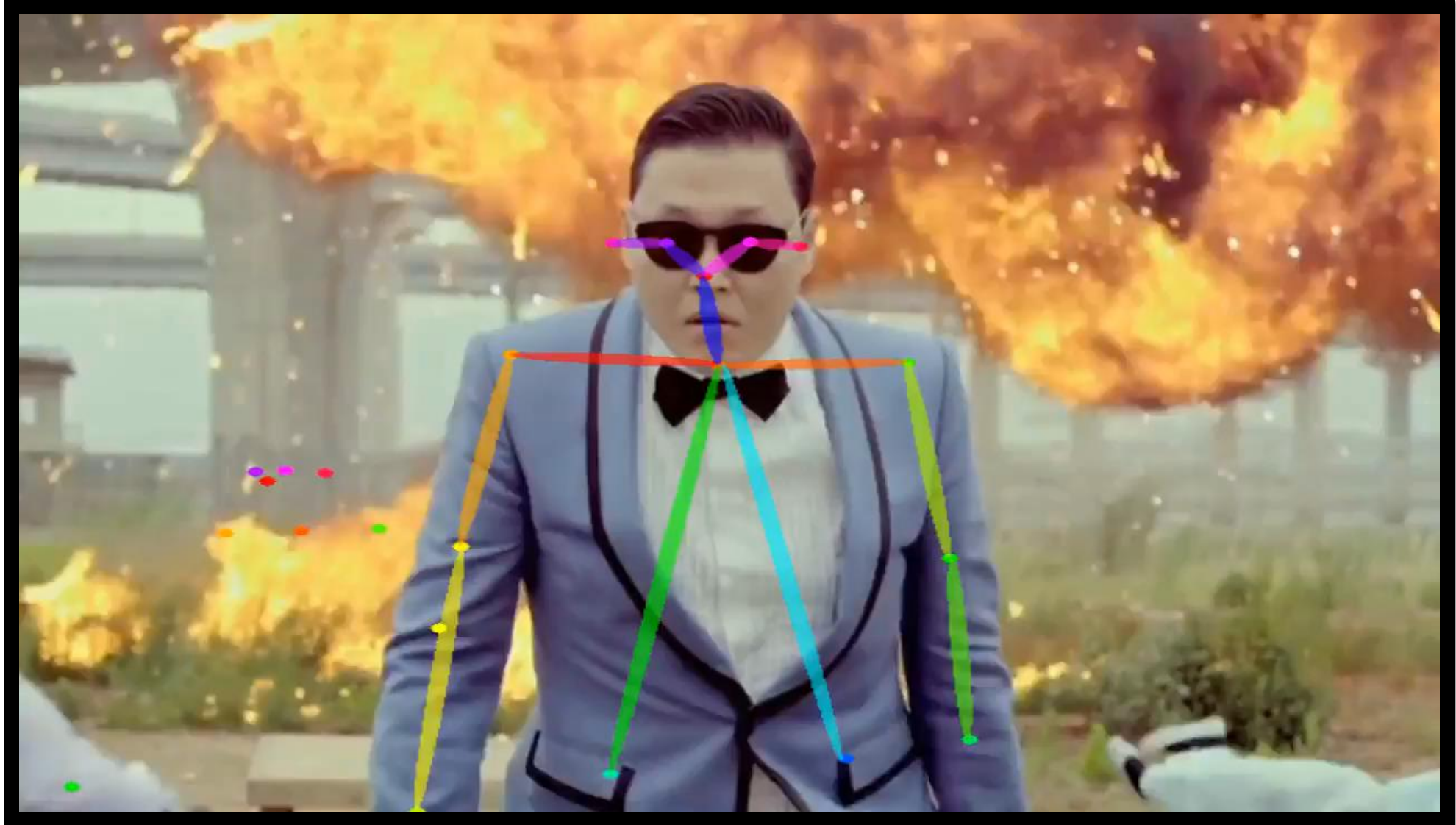
3) Model selection & Training



The background of the slide is a dark blue gradient. It features a large, semi-transparent circle on the right side and a vertical semi-transparent bar on the left side. The text "Session Break" is centered in the middle of the slide in a white, sans-serif font.


Session Break

HUMAN POSE ESTIMATION



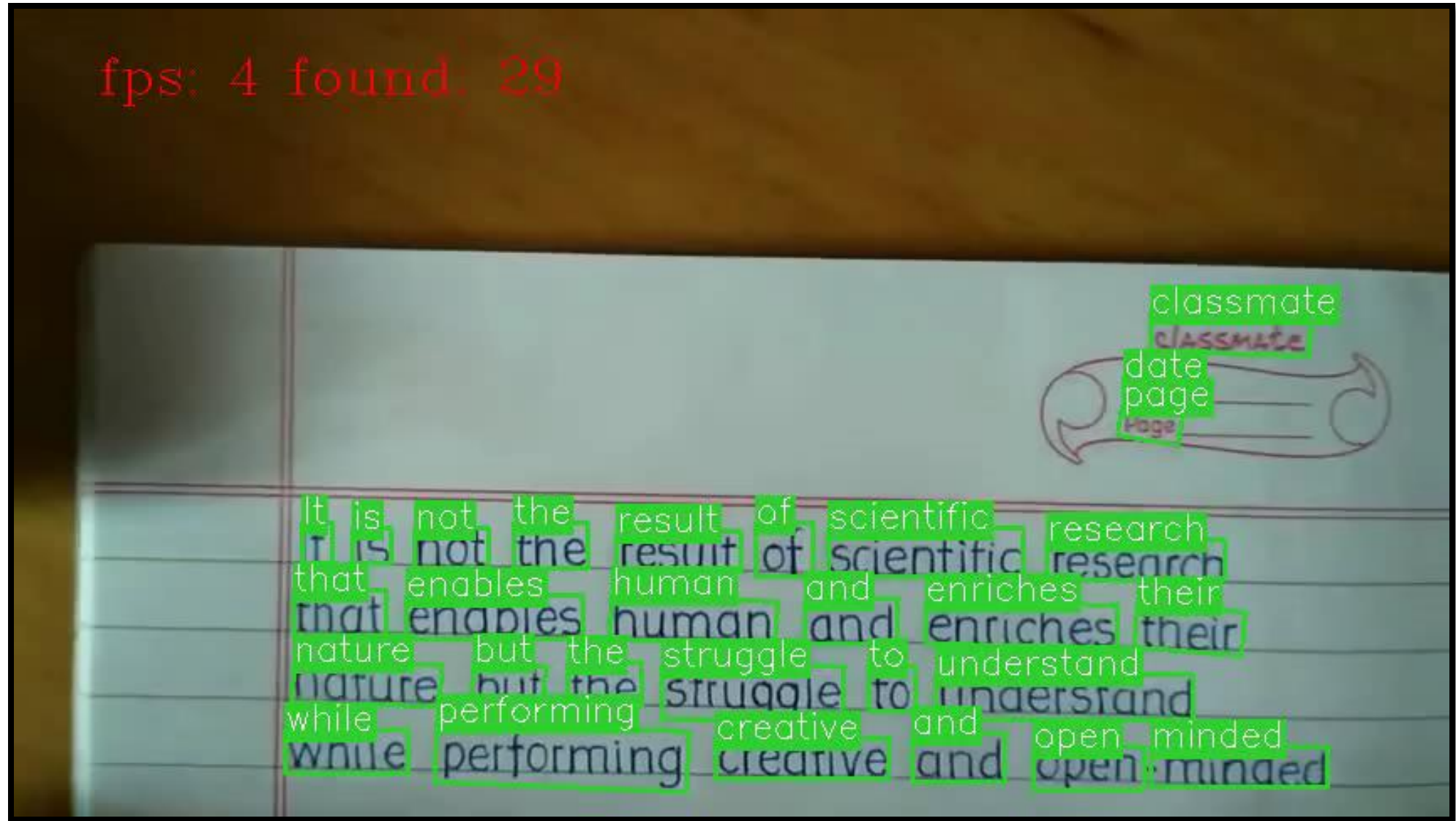
INSTANCE SEGMENTATION

Inference time: 117.521 ms
OpenCV rendering time: 0.000 ms

A large, solid blue rectangle occupies the majority of the lower half of the image. It is framed by a thin black border. In the top-left corner of this blue area, there is text indicating performance metrics: 'Inference time: 117.521 ms' in blue and 'OpenCV rendering time: 0.000 ms' in red.

Optical Character Recognition

fps: 4 found: 29





Thank you
