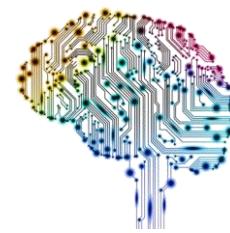
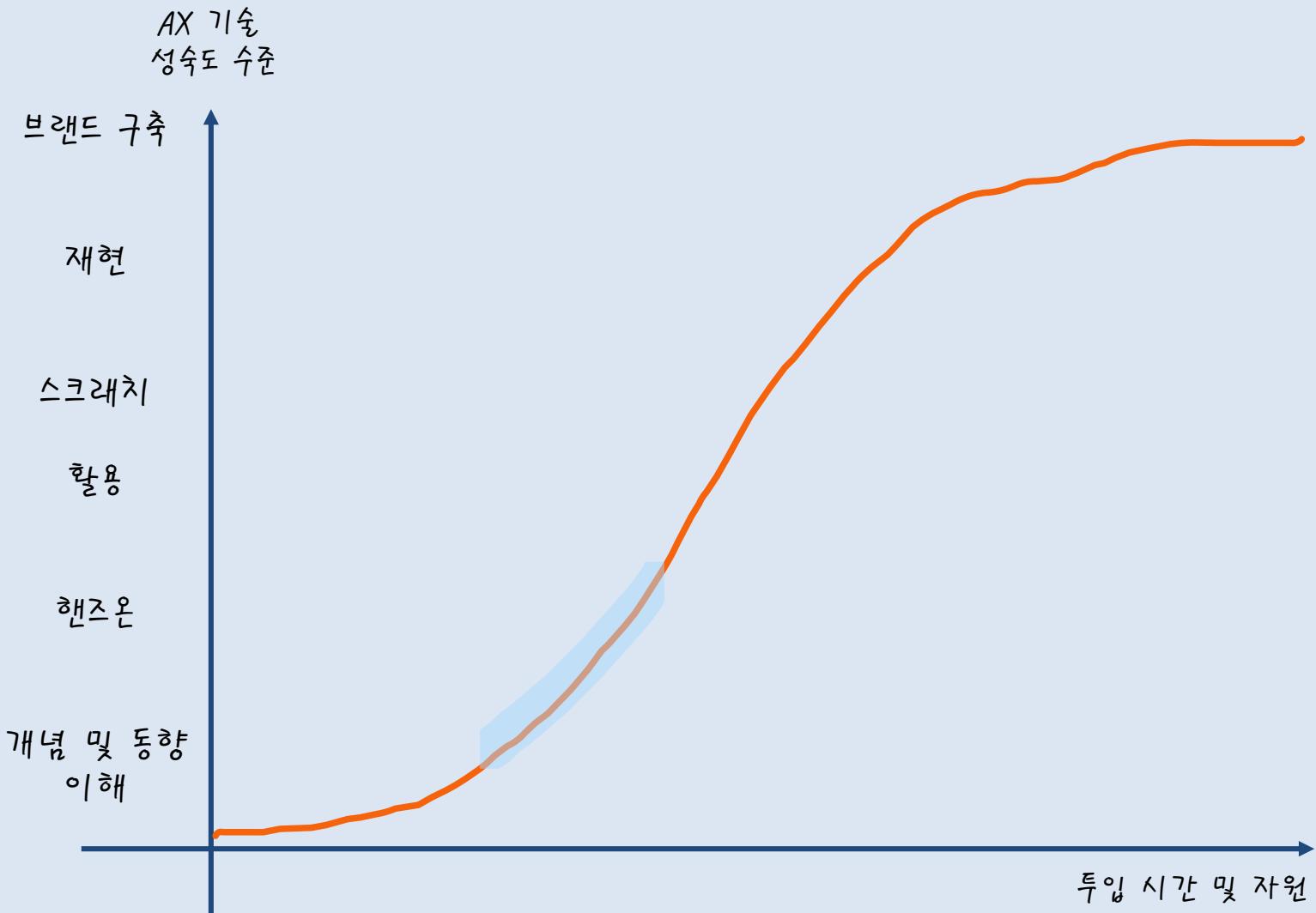


# AX foundation and Trend 2025

Kang Taewook  
laputa9999@gmail.com  
<https://daddynkidsmakers.blogspot.com>



2025/7

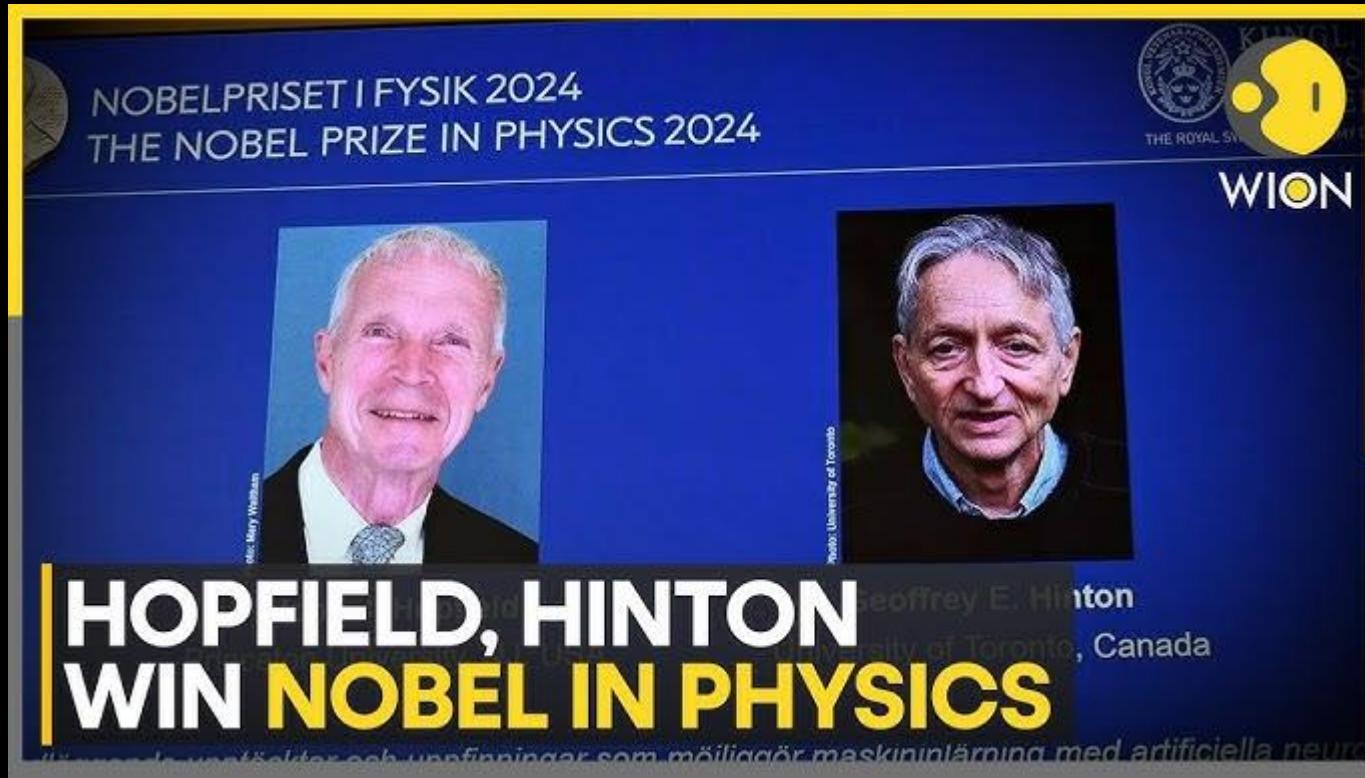


# 컨텐츠

AX 트랜드와 서비스  
머신러닝 기초 개념  
딥러닝의 구조적 이해  
자연어 처리  
Transformer와 최신 AI 모델  
LLM 과 AI Agent  
AI 모델의 서비스화 개요  
AX전략

# AX 트랜드와 서비스

AX 트랜드  
AI vs 일반 소프트웨어 구조 비교  
AI 서비스 개발 흐름



Members of the Nobel Committee for Chemistry at the Royal Swedish Academy of Sciences explain the work of 2024 Nobel Prize in Chemistry winners David Baker, Demis Hassabis and John M. Jumper.JONATHAN NACKSTRAND/AFP via Getty Images

AI Pioneers Geoffrey Hinton And John Hopfield Win Nobel Prize For Physics | Latest News | WION

# AX 트랜드

● Live



# AX 트랜드

We've been testing the capabilities of Gemini, our new multimodal AI model.

We've been capturing footage to test it on a wide range of challenges, showing it a series of images, and asking it to reason about what it sees.

## Large Language Models as General Pattern Machines

Suvir Mirchandani Fei Xia Pete Florence Brian Ichter  
Danny Driess Montserrat Gonzalez Arenas Kanishka Rao  
Dorsa Sadigh Andy Zeng

[general-pattern-machines.github.io](https://general-pattern-machines.github.io)



TITLE	CITED BY	YEAR
<a href="#">Large language models as general pattern machines</a> S Mirchandani, F Xia, P Florence, B Ichter, D Driess, MG Arenas, K Rao, ... arXiv preprint arXiv:2307.04721	103	2023

# AX 트랜드

## 생성 AI

The screenshot displays a desktop interface with several open windows:

- File Explorer:** Shows an image file named "zaha\_ddp.jpg".
- Terminal Window:** Titled "명령 프롬프트 - ollama run llama3.2-vision", it shows command-line options and environment variables. It also displays a prompt and its JSON response about a modern building.
- Web Browser:** Titled "KREA | Chat", it shows a URL "https://www.krea.ai/chat". The page content includes a description of a modern building and a generated image preview.
- KREA Chat Interface:** A detailed description of a modern, all-white building with curved lines and a large open space, featuring a staircase and a glass wall. Below this is a generated image preview and various editing tools like "Change style" and "Change background".

Red arrows highlight specific parts of the terminal output and the generated image description in the KREA interface.

```
--format string      Response format (e.g. json)
--h, --help           help for run
--insecure           Use an insecure registry
--keepalive string   Duration to keep a model loaded (e.g. 5m)
--nowordwrap         Don't wrap words to the next line automatically
--verbose            Show timings for response

Environment Variables:
OLLAMA_HOST          IP Address for the ollama server (default 127.0.0.1:11434)
OLLAMA_NOHISTORY     Do not preserve readline history

F:\projects\ollama>ollama run llama3.2-vision
>>> image: ./zaha_ddp.jpg
Added image: ./zaha_ddp.jpg
The image depicts a modern, white building with a sleek and minimalist design. The purpose of the image is to showcase the architectural features of the building.

* A large, open space in a building:
+ The space is well-lit, with natural light pouring in through the windows.
+ The floor is made of polished concrete or marble, reflecting the light and adding to the sense of openness.
+ There are no visible walls or partitions, creating a sense of freedom and flexibility.
* A staircase leading up to the second floor:
+ The staircase is curved and sleek, with a minimalist handrail that adds to the overall aesthetic.
+ The steps are made of the same material as the floor, creating a sense of continuity.
+ The staircase appears to be the main entrance to the building, drawing attention upwards.
* A glass wall on the left side:
+ The glass wall is floor-to-ceiling, providing an unobstructed view of the surrounding area.
+ The wall is framed by a thin metal strip, adding a touch of sophistication.
+ The glass wall appears to be a design feature, rather than a functional one, as it does not seem to serve any practical purpose.

Overall, the image showcases a modern and sleek building with an emphasis on openness and minimalism. The use of natural light, polished floors, and curved lines creates a sense of fluidity and freedom.

>>> 한글로 설명해
이미지는 현대적인 건물의 내부를 보여주고 있습니다. 건물은 백색을 기본으로 하고 있으며, 정형화된 디자인 특징을 띠고 있습니다.

* 건물 내의 대공간:
+ 공간은 자연광이 잘 들여다보이며, 창문이 많은 편입니다.
+ 바닥에 광택이 난 청크리트나 마루블로 구성되어 있으며, 이를 통해 자연광과 공간을 더 연결시킵니다.
+ 벽이나 패티션 없이 넓고 깔끔한 느낌의 공간입니다.

* 두 번째 층으로 올라가는 계단:
+ 계단은 목재나 철재로 제작되었으며, 손잡이도 간결한 디자인으로 이루어져 있습니다.
+ 계단의 각 단계는 바닥과 같은 재질로 구성되어 있으며, 이를 통해 걸음을 내에 일관된 느낌을 줄 수 있습니다.
+ 두 번째 층으로 올라가는 계단은 계단을 중심으로 출렁거리는 듯한 디자인입니다.

* 왼쪽 측면에 있는 유리 벽:
+ 유리 벽은 바닥부터 천장까지 이어져 있으며, 건물 외부와 무제한적인 시야를 제공합니다.
+ 창문은 금속으로 이루어진 프레임이 벽의 경계를 이루고 있습니다.
+ 유리 벽은 기능적 요소가 아닌 예술적인 요소로 보이며, 이는 일체감이나 공간을 풍부하게 만들기 때문입니다.

전체적으로 이미지 내에 있는 건물을 현대적인 건축 양식을 띠며, 공간의 개방성과 간결한 디자인을 강조하고 있습니다. 자연광, 광택이 난 바닥, 쪽면으로 제작된 계단 등이 이를 증명합니다.

>>> 미드저니에 이미지 생성 목적의 프롬프트로 만들어줘
Here is a midjourney prompt that describes the image:

"Image of a modern, all-white building with curved lines and a large open space. The building features a sleek and minimalist design, with a staircase leading up to the second floor and a glass wall on the left side. The interior is well-lit, with natural light pouring in through the windows.">>> Send a message (/? for help)
```

# AX 트랜드

생성 AI



Krea | Video



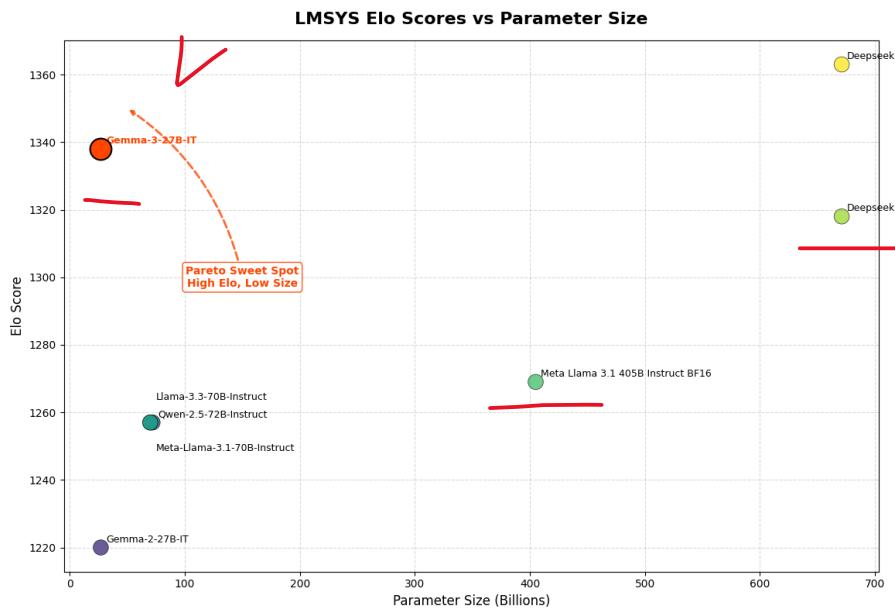
“Make this look like a cozy summer evening in a Ghibli film — soft golden light, fireflies glowing, and a gentle breeze moving the grass.”

“Turn the background into a Ghibli-style village with tiny tea shops, lanterns, and warm, inviting streets.”

# AX 트랜드

## Gemma 3

구글이 개발해 2025년 3월 10일에 출시한 LLM(1B, 4B, 12B, 27B). 차세대 경량 오픈 멀티모달 AI 모델로, 텍스트와 이미지를 동시에 처리할 수 있는 기능 지원. 컨텍스트 크기는 32k(1B), 128k(4B-27B) 지원

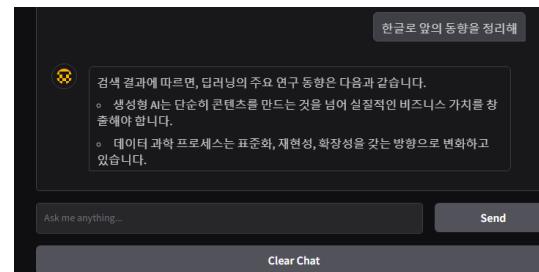


### Gemma 3 Release - a google Collection



What is the password?

Based on the image, the password is **aaeu**



Daddy Makers:  
Gemma3 기반 Ollama  
활용 AI 에이전트 개발  
핵심 Function Call 구현해보기

# AX 트랜드

AI 과학자

[Sakana AI](#) Scientist는 과학 연구 전 과정을 자동 수행하는 시스템

This is a maze  
navigation task.

Watch the CTM  
observe and figure out  
its way as we give it

## Time to Think

# AX 트랜드

## MCP

The image is a composite of three screenshots. On the left, the Figma interface shows a wireframe project titled 'Wireframing in Figma'. The central part shows a context menu with 'Copy/Paste as' highlighted. On the right, a landing page for 'BrandName' features a purple header with the text 'Build Amazing Digital Experiences' and a subtext 'Create stunning websites and applications with our modern design system. From wireframes to production-ready code.' Below the header are two buttons: 'Start Building' and 'Watch Demo'. At the bottom, there are three metrics: '6.1K+' under 'Happy Customers', '6.1%' under 'Uptime', and '24/7' under 'Support'.

Daddy Makers: 인공지능 AI 에이전  
트 표준 프로토콜 MCP 개념, 사용,  
개발 및 동작 구조 분석하기

## Why Choose Us

We provide cutting-edge solutions that help your business grow and thrive in the digital landscape

# AX 트랜드

## 노코드(No-code)



# AX 트랜드

China 이니셔티브



### Structured 3D Latents for Scalable and Versatile 3D Generation

Jianfeng Xiang<sup>1,3\*</sup> Zelong Lv<sup>2,3\*</sup> Sicheng Xu<sup>3</sup> Yu Deng<sup>3</sup> Ruicheng Wang<sup>2,3\*</sup>  
Bowen Zhang<sup>2,3\*</sup> Dong Chen<sup>3</sup> Xin Tong<sup>3</sup> Jiaolong Yang<sup>3†</sup>

<sup>1</sup>Tsinghua University <sup>2</sup>USTC <sup>3</sup>Microsoft Research

<https://trellis3d.github.io>

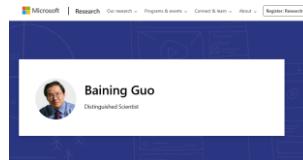
2 Dec 2024



I am currently a Ph.D student majoring in Computer Science at Tsinghua University (THU) under the supervision of Prof. Baining Guo. Besides, I have been a research intern at Microsoft Research Asia (MSRA) since Jul 2021, working closely with Sicheng Xu, Jiaolong Yang and Xin Tong.

Before that, in Jun 2022, I received my B.E. in Electronic Information Engineering at University of Science and Technology of China (USTC) from the School of the Gifted Young and the School of Information Science and Technology.

[Jianfeng Xiang | Home](#)



[Baining Guo at Microsoft Research](#)

[Daddy Makers: 오픈소스 기반 2D Image to 3D model 소개](#)

# AX 트랜드

[All](#) [Images](#) [Videos](#) [Shopping](#) [Books](#) [News](#) [Maps](#) [More](#) [Tools](#)

 머니투데이

### "추격조'에 GPU 몰아주면 韓 딥시크 10개 나와"

"글로벌 AI 추격조를 만들어 '3년간 한국 데이터를 다 가져다 써라' 해야 합니다. 정부가 연내 GPU를 1만개를 확보해 상하반기 각 5개 기업에 2000..."

Feb 6, 2025



 한국일보

### [현장] "GPU·데이터 몰아주면 한국에서도 딥시크 10개 나온다"

6일 과학기술정보통신부가 개최한 중국 인공지능(AI) 모델 '딥시크'의 등장 이후 국내 AI 산업의 경쟁력을 점검하는 회의에서 거대언어모델(LLM)을...

Feb 7, 2025



 중앙일보

### 韓 AI 회사 "지금이 오히려 기회...딥시크 10개 만들 방법 있다" [팩플]

중국 생성 인공지능(AI) 모델 딥시크의 등장은 한국 AI 산업에 위기일까 기회일까. 국내 AI 기업들은 'AI 추격조'를 만드는 등 정부의 전폭적 지원이...

Feb 6, 2025



 노컷뉴스

### [Q&A]AI 강국되려면 GPU 잡아라..."품귀현상에도 구할 수 있다"

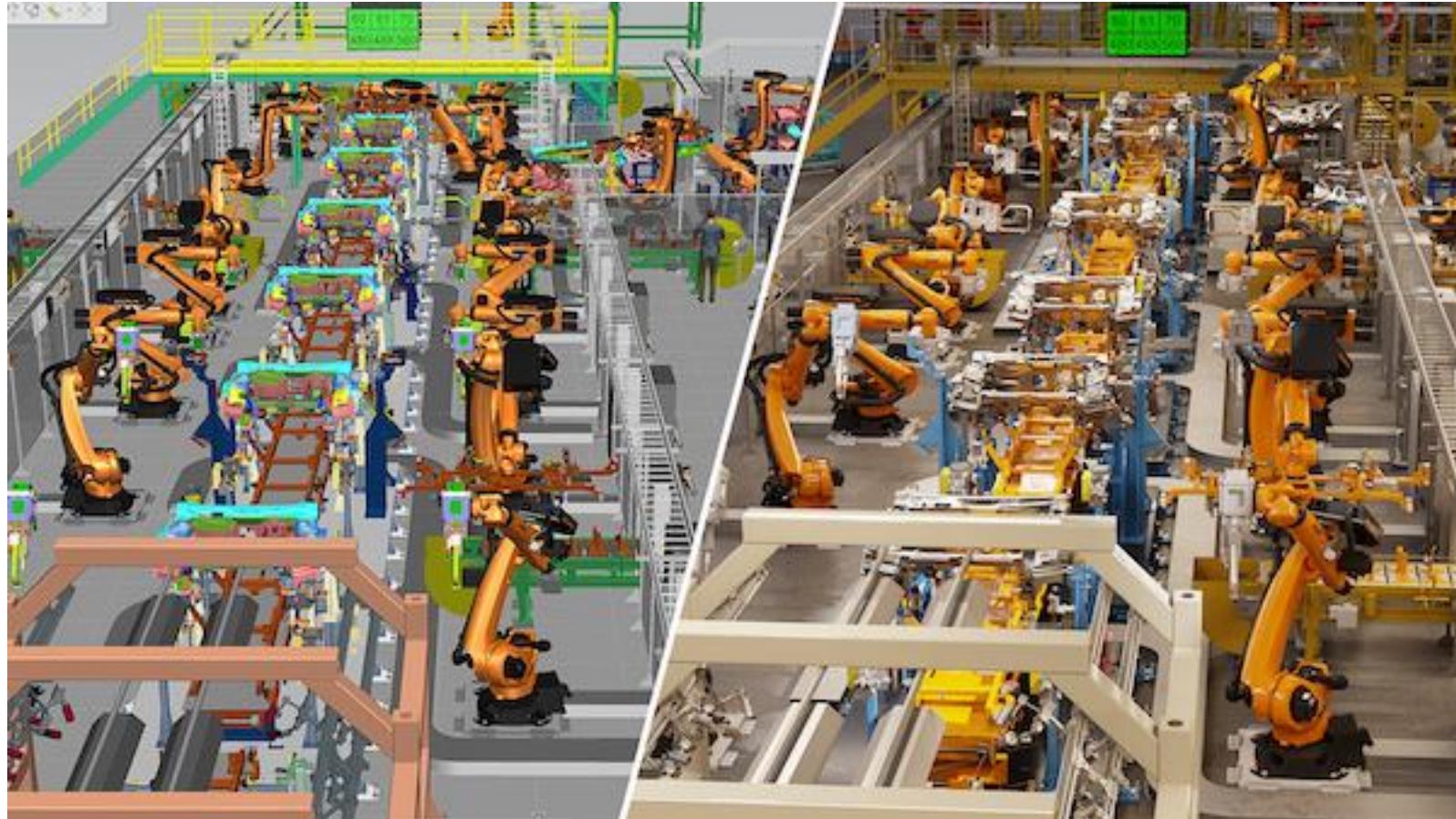
"더 많은 인프라를 가지고 양질의 데이터를 확보해서 AI 모델을 만드는 것이 중요하다" (배경훈 LG AI연구원장) "GPU 1만 개 확보해서 10개 회사에..."

1 month ago



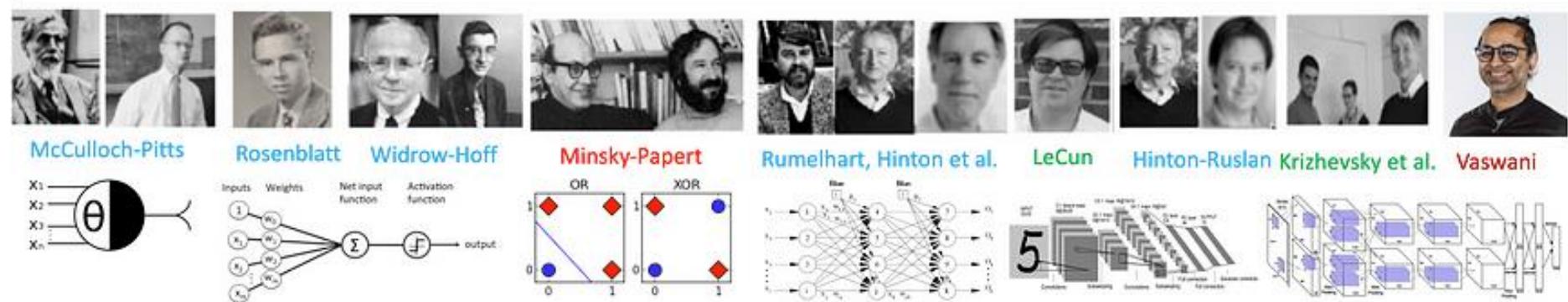
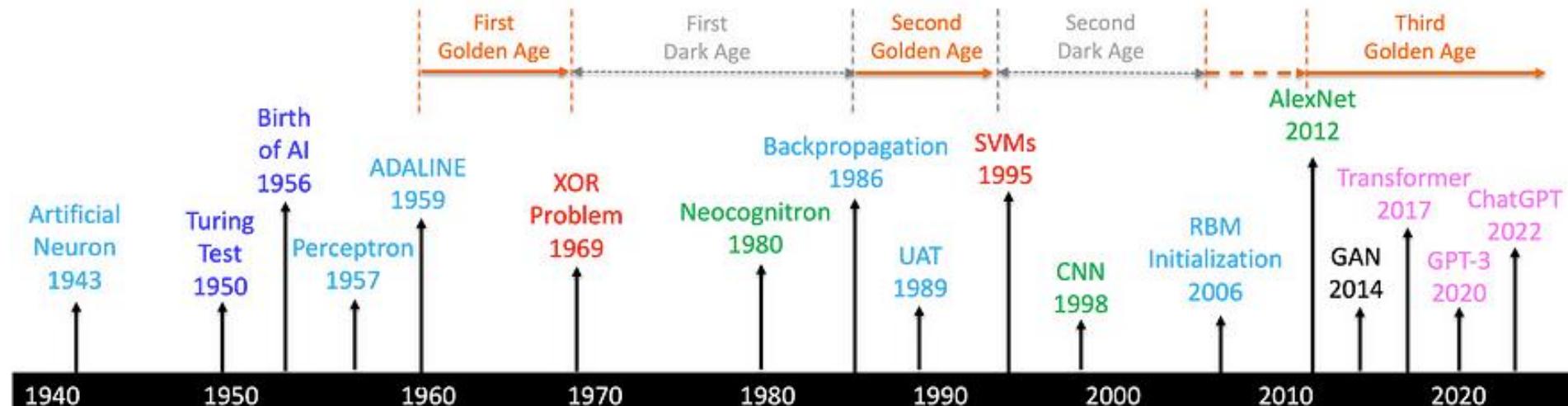
# AX 트랜드

## 디지털트윈 + AX + 로보틱스



Siemens Process Simulate (left) connects to NVIDIA Omniverse (right) to enable a full-design-fidelity, photorealistic, real-time digital twin.  
[https://www.robotics247.com/article/siemens\\_xcelerator\\_nvidia\\_omniverse\\_accelerate\\_digital\\_twins\\_manufacturing](https://www.robotics247.com/article/siemens_xcelerator_nvidia_omniverse_accelerate_digital_twins_manufacturing)

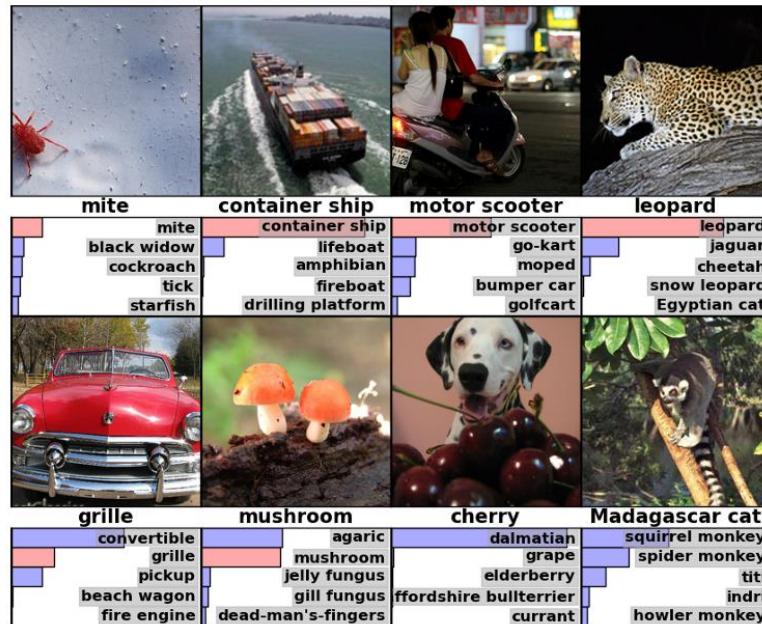
## A Brief History of AI with Deep Learning



# AI 서비스

## AI vs 일반 소프트웨어 구조

Convolutional neural networks running on GPUs (2012) Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, Advances in Neural Information Processing Systems 2012



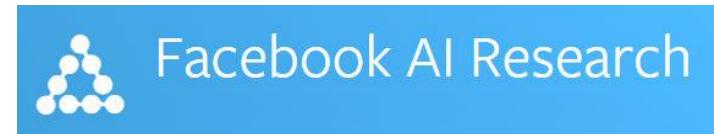
Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
<i>SIFT + FVs [7]</i>	—	—	26.2%
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	16.4%
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	15.3%

[ImageNet Classification with Deep Convolutional Neural Networks](#)

Deep CNN, ReLU, Dropout



2013. Google acquired DNN



Yann LeCun.  
Chief AI Scientist for Facebook AI Research (FAIR)

# Computer Vision & Learning Group

[Home](#)[People](#)[Publications](#)[Research](#)[Teaching](#)[Open Positions](#)

Prof. Dr. Björn Ommer

Contact

University of Munich

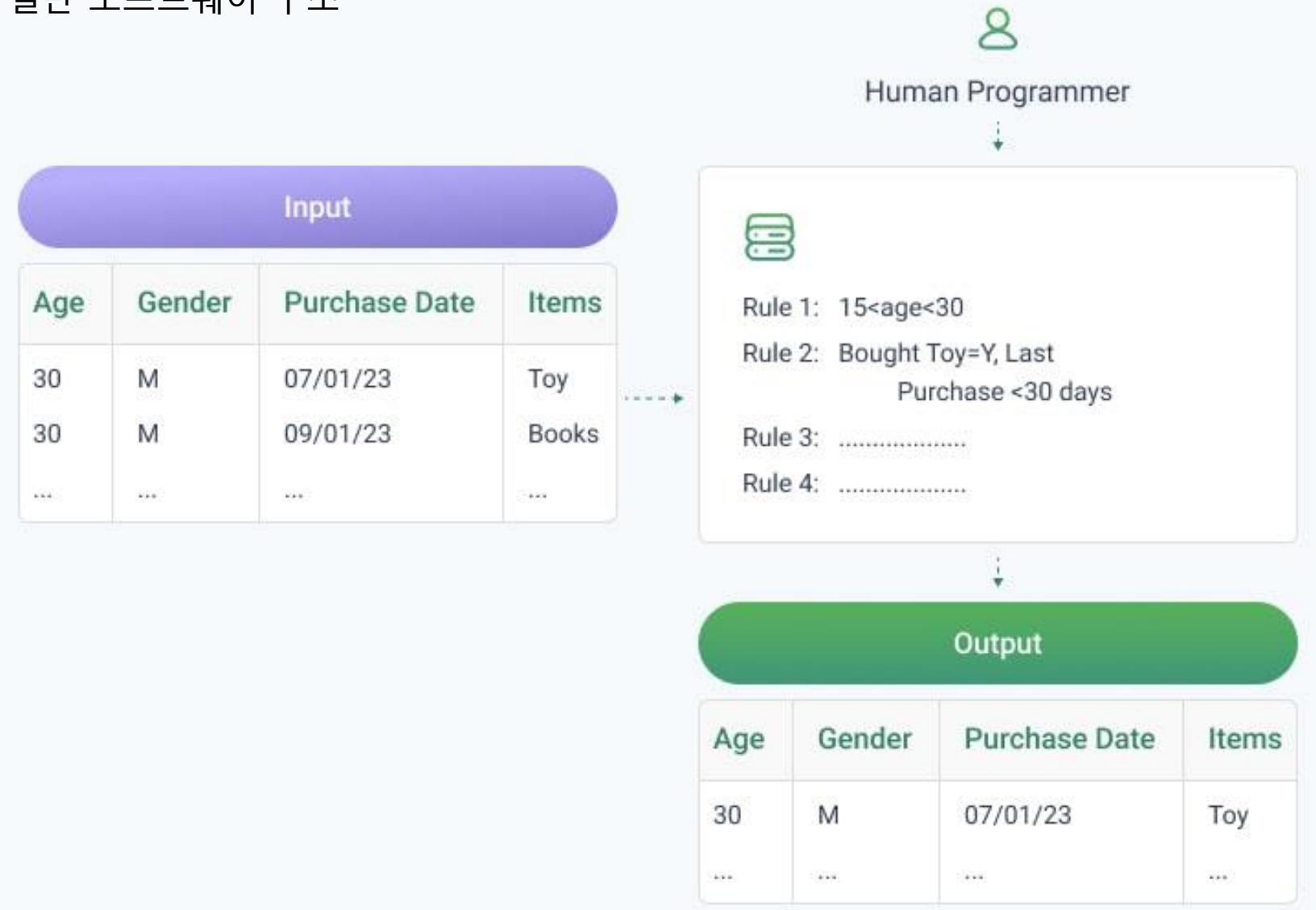


Stable Diffusion Developer [Computer Vision & Learning Group \(ommer-lab.com\)](http://ommer-lab.com)

# AI 서비스

## AI vs 일반 소프트웨어 구조

### 일반 소프트웨어 구조



# AI 서비스

## AI vs 일반 소프트웨어 구조

AI

Historical Purchase Data

Age	Gender	Purchase Date	Items
30	M	07/01/23	Toy
30	M	09/01/23	Books
...	...	...	...

Input New Unseen Data

Age	Gender	Items
35	F	Sportswear

Learning  
Algorithm

Model



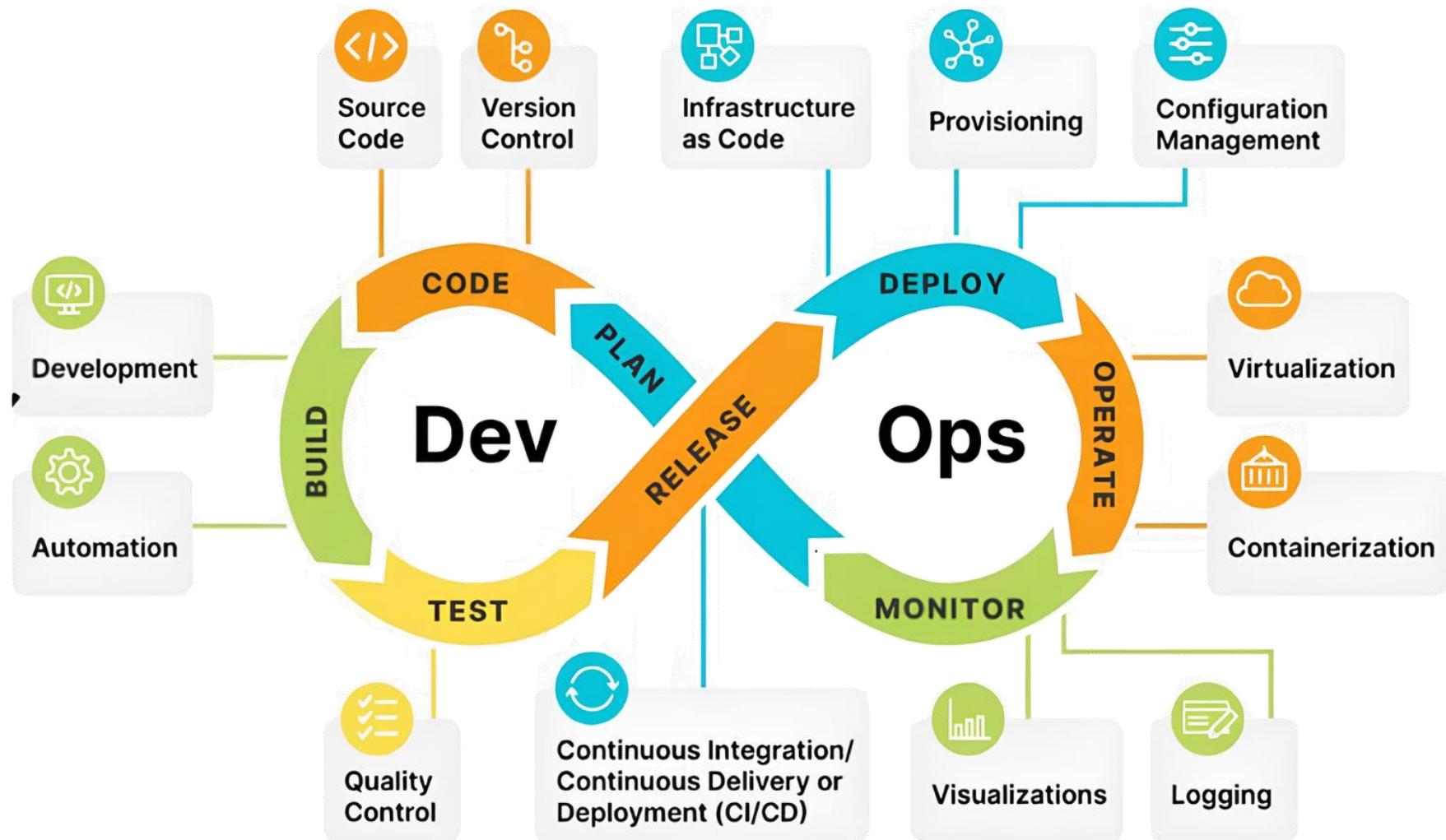
Prediction/ Output

# AI 서비스

## AI vs 일반 소프트웨어 구조

구분	일반 소프트웨어	AI 서비스
핵심 로직	개발자가 직접 코딩한 명시적 규칙과 알고리즘	데이터로부터 학습된 패턴 (모델)
동작 방식	<b>결정론적 (Deterministic)</b> 동일한 입력에 대해 항상 동일한 결과를 출력	<b>확률적 (Probabilistic) &amp; 적응적 (Adaptive)</b> 학습 데이터와 모델 상태에 따라 결과가 달라질 수 있으며, 새로운 데이터로 지속적 성능 개선 가능
개발 과정	요구사항 분석 → 설계 → 코딩(구현) → 테스트	데이터 수집/가공 → 모델 학습 및 평가 → 서빙 → 모니터링 및 재학습
데이터 역할	시스템이 처리해야 할 입력값	모델을 학습시키고 검증하는 핵심 재료
유지보수	버그 수정, 기능 추가 등 코드 업데이트	성능 저하(Drift) 모니터링, 모델 재학습, 데이터 편향(Bias) 관리
장점	<ul style="list-style-type: none"><li>- 예측 가능성 및 신뢰성이 높음</li><li>- 로직이 명시적이므로 디버깅 및 검증이 용이</li><li>- 요구사항이 명확하면 개발 기간과 비용 예측 용이</li></ul>	<ul style="list-style-type: none"><li>- 복잡하고 확률적 비정형적인 문제 해결</li><li>- 데이터에 스스로 적응/학습/성능 향상</li><li>- 인간 직관이 필요한 패턴 인식. 개인화 강점</li></ul>
단점	<ul style="list-style-type: none"><li>- 예상치 못한 데이터나 변화에 대응 난해</li><li>- 규칙을 정의할 수 없는 문제(예. 이미지 인식) 는 해결이 불가능</li><li>- 로직이 복잡해질수록 개발 유지보수가 기하급수적으로 어려워짐</li></ul>	<ul style="list-style-type: none"><li>- '블랙박스' 문제: 왜 그런 결정을 내렸는지 설명하기 어려움</li><li>- 데이터 의존성: 대량의 고품질 데이터가 필수적이며, 데이터 편향에 취약</li><li>- 학습 및 운영에 높은 컴퓨팅 비용과 전문 인력이 필요</li><li>- 환각현상. 100% 정확성을 보장할 수 없음</li></ul>

## DevOps



# AI 서비스

AI 서비스 개발 흐름: 데이터 → 모델 → 서비스화

## DevOps

단계	핵심 활동	상세 설명
Plan	요구사항 정의, 기능 명세화, 작업 계획 및 주제 등의 활동을 수행함.	프로젝트의 목표와 요구사항을 정의하고 개발 우선순위를 결정하는 초기 단계임.
Code	소스 코드 작성과 Git 등을 이용한 버전 관리를 통해 협업을 진행함.	계획에 따라 실제 코드를 작성하고, 변경 이력을 관리하며 협업하는 단계임.
Build	개발 환경 통합과 빌드 자동화를 통해 소스 코드를 실행 파일로 변환함.	작성된 소스 코드를 컴파일하고 종속성을 관리하여 실행 가능한 소프트웨어로 자동 변환하는 단계임.
Test	품질 관리와 단위 및 통합 테스트 자동화를 통해 소프트웨어의 안정성을 검증함.	빌드된 소프트웨어에 버그가 없는지, 요구사항을 충족하는지 자동으로 검증하여 품질을 보장하는 단계임.
Release	릴리스 노트 작성, 버전 관리, 배포 준비 등의 활동을 포함함.	테스트를 통과한 버전을 실제 운영 환경에 배포 할 수 있도록 준비하고 최종 승인하는 단계임.
Deploy	프로비저닝(Provisioning), 코드로 인프라 관리(IaC), 구성 관리(Configuration Management) 통해 배포환경 구축 자동화.	프로비저닝은 사용자 요구에 맞게 IT 인프라(서버, 네트워크, 스토리지 등) 생성, 설정, 사용할 수 있게 준비하는 과정. 앱을 프로덕션 서버에 배포함
Operate	가상화(Virtualization) 및 컨테이너화(Containerization) 기술을 활용하여 인프라를 운영함.	배포된 서비스가 안정적으로 구동되도록 서버, 네트워크 등 인프라를 관리하고 지원하는 단계임.
Monitor	성능 지표 시각화(Visualizations)와 시스템 이벤트 로깅(Logging)을 수행함.	서비스의 성능과 상태를 지속적으로 관찰하고, 발생한 문제나 개선점을 다음 계획 단계에 피드백 하는 단계임.
CI / CD	지속적 통합(CI)과 지속적 제공 및 배포(CD)를 구현하는 것임.	DevOps의 핵심 엔진으로서, 코드 통합부터 배포 까지 전 과정을 자동화하여 전체 사이클을 원활하게 연결하는 개념임.

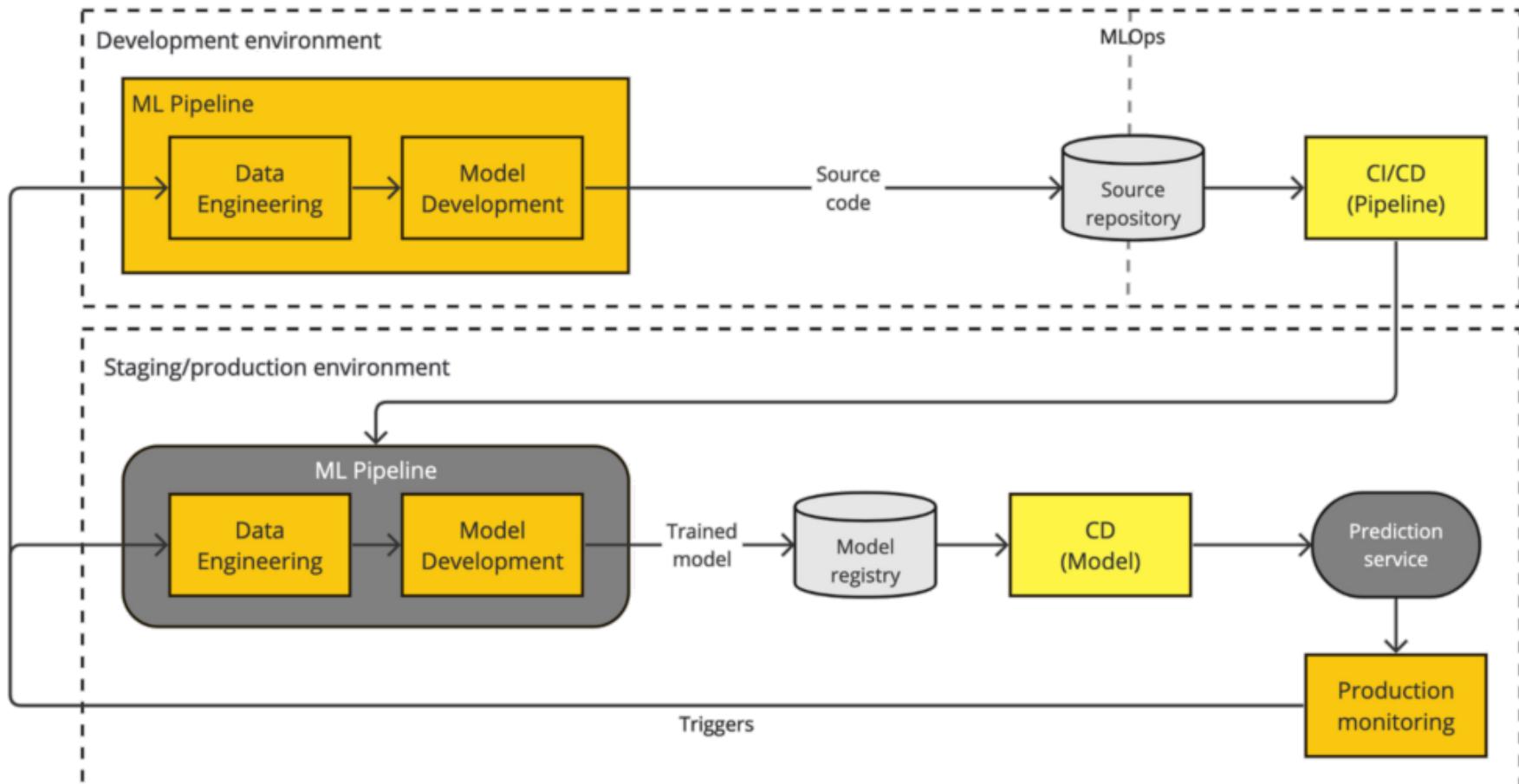
# AI 서비스

AI 서비스 개발 흐름: 데이터 → 모델 → 서비스화

## MLOps

AI 솔루션 개발 생산성을 촉진하고 데이터 품질을 유지하는 데 효과적인 접근 방식

- 머신러닝 엔지니어가 모델 개발 및 배포 속도를 높일 수 있도록 지원하는 인프라
- ML 모델을 지속적으로 모니터링하고 검증하는 과정이 필수적
- 지속적인 통합/배포/학습(CI/CD/CT)의 순환 주기를 완성하는 것이 중요



# AI 서비스

AI 서비스 개발 흐름: 데이터 → 모델 → 서비스화

## MLOps

단계	핵심 활동 (Key Activities)	상세 설명
Problem Framing	비즈니스 목표 이해, 성공 지표 정의, 데이터 가용성 평가 등을 수행함.	비즈니스 문제를 머신러닝 문제로 변환하고, 프로젝트의 성공 기준과 방향을 설정하는 초기 기획 단계.
Data Engineering	데이터 수집, 정제, 레이블링, 특징 공학(Feature Engineering), 데이터 버전 관리 등을 진행함.	모델 학습에 필요한 데이터를 모으고, 정제하여 품질을 높이며, 모델이 이해할 수 있는 특징(Feature)으로 가공.
Model Engineering	알고리즘 선택, 모델 코드 작성, 하이퍼파라미터 튜닝, 모델 성능 검증, 실험 관리 등을 수행함.	준비된 데이터로 모델을 학습시키고, 다양한 지표를 통해 성능을 객관적으로 평가하여 최적의 모델을 찾는 핵심 단계.
Model Deployment	학습된 모델을 패키징하고, 서빙 인프라(API 서버, 클라우드 환경 등)에 배포할 수 있도록 준비함.	검증이 완료된 모델을 실제 사용자가 접근할 수 있는 운영 환경으로 이전.
Serving & Inference	배포된 모델을 통해 실시간 또는 배치(Batch) 형태로 예측 결과를 제공하고, API를 통해 다른 서비스와 연동함.	실제 사용자 데이터에 대해 모델이 예측을 수행하며, AI 서비스의 핵심 기능을 제공.
Model Monitoring	모델 성능 저하(Drift), 데이터 편향성(Bias), 시스템 자원 사용량 등을 지속적으로 추적하고 기록함.	운영 중인 모델이 예측 성능을 일관되게 유지하는지, 예상치 못한 오류는 없는지 실시간으로 감시.
Retraining	모니터링 결과를 바탕으로 새로운 데이터로 모델을 다시 학습시키고, 필요시 새로운 파이프라인을 가동함.	시간이 지나면서 변화하는 데이터 패턴에 모델이 적응하도록 하여, 서비스의 전반적인 성능과 정확성을 지속적으로 유지하고 개선하는 과정.

# AI 서비스

AI 서비스 개발 흐름: 데이터 → 모델 → 서비스화

## MLOps

단계 (Phase)	솔루션 예시
Problem Framing	자체 분석 도구, 스프레드시트, Nocode, 바이브 코딩 서비스
Data Engineering	PyTorch, Kafka, Spark, Pandas, Scikit-learn, NumPy
Model Engineering	TensorFlow, PyTorch, Keras, Scikit-learn, MLflow, Weights & Bias
Model Deployment	ONNX((Open Neural Network Exchange), Docker, Kubernetes, Tens orFlow Serving, TorchServe, FastAPI, Flask
Serving & Inference	TensorFlow Serving, TorchServe, NVIDIA Inference Server, AWS Sag eMaker Inference, Google AI Platform Prediction, Azure Machine Lear ning Endpoints
Model Monitoring	Prometheus, Grafana
Retraining	Kubeflow Pipelines, MLflow Projects, AWS SageMaker Pipelines, Goo gle Cloud AI Pipelines, Azure Machine Learning Pipelines

AI가 세계를 지배할 거라는  
AI 알못들:



내가 만든 AI:



# 머신러닝 기초 개념

AI학습 및 개발 환경  
분류와 회귀의 차이

Overfitting, 정규화, 검증셋 개념

주요 알고리즘(Decision Tree, KNN 등) 개요  
Scikit-learn을 활용한 파이프라인 예시 소개

# 머신러닝 기초 AI학습 및 개발 환경

Python (라이브러리 호환성 고려해 3.11버전 이상 권장. 2025.7 시점)

Python 다운로드

Mac 사용자: Mac에 Python 설치 가이드

설치 후 터미널에서 python --version 로 실행이 제대로 되는지 확인해 볼 것

NVIDIA 드라이버 (NVIDIA GPU 사용 시)

NVIDIA 드라이버 다운로드

설치 후 터미널에서 nvidia-smi로 확인

CUDA Toolkit (NVIDIA GPU 사용 시)

CUDA Toolkit 다운로드

설치 시 GPU 및 드라이버 호환성 확인

환경 변수에 CUDA 경로 추가

**Github 도구 설치**

다음 링크에서 도구 설치함.

<https://docs.github.com/ko/desktop/installing-and-authenticating-to-github-desktop/installing-github-desktop>

**Anaconda (버전 24.0 이상 권장)**

다음 링크에서 도구 설치함.

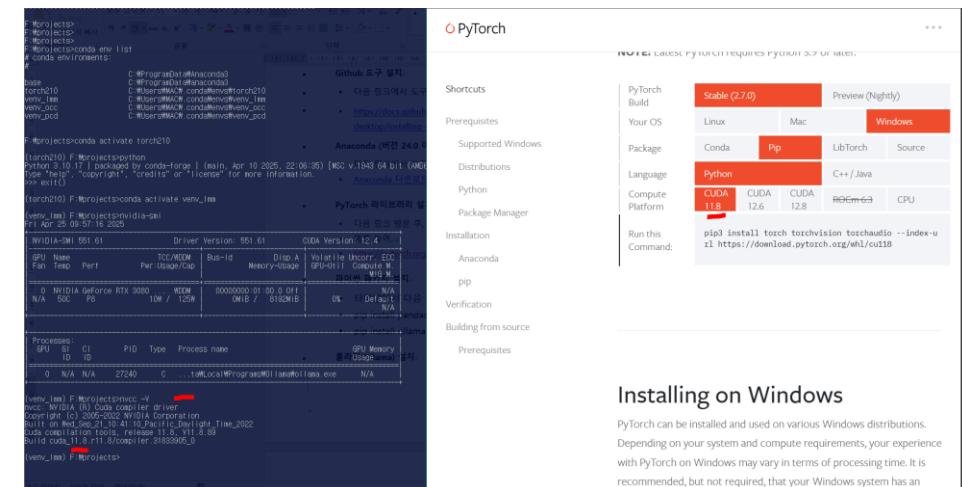
Anaconda 다운로드

**PyTorch 라이브러리 설치**

다음 링크 방문 후, CPU 버전 혹은 GPU 드라이버 버전에 맞게 터미널에서 설치함

<https://pytorch.org/get-started/locally/>

CPU의 경우, 다음과 같이 터미널에서 명령을 입력해 설치함.



# 머신러닝 기초 AI학습 및 개발 환경

## 도커(Docker) 설치(옵션)

가상이미지 기반 작업을 위해서는 설치가 필요합니다.

<https://www.docker.com/get-started/> 방문 설치

## 올라마(Ollama) 설치

로컬 LLM AI 도구를 사용하려면 Ollama 설치가 필요합니다.

<https://www.ollama.com/> 에 방문해 설치

## 코드 편집기 및 IDE 설치

<https://www.sublimetext.com/> 설치

<https://code.visualstudio.com/download> 설치. 상세한 설치법은

## 클로드 Desktop 설치

<https://claude.ai/download> 설치

<https://docs.github.com/ko/desktop/installing-and-authenticating-to-github-desktop/installing-github-desktop>

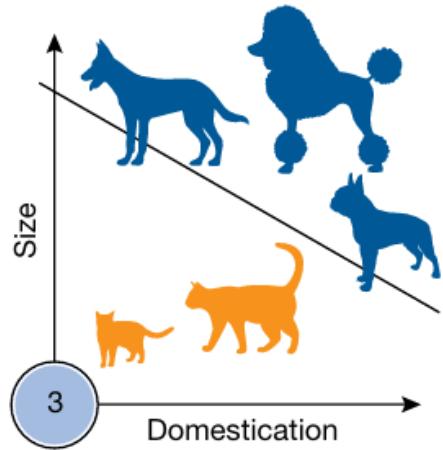
# 머신러닝 기초 AI학습 및 개발 환경

- Colab Pro 가입(유료): <https://colab.research.google.com/signup>
- ChatGPT 가입(유료)
- ChatGPT API Pay as you go 가입(유료. 한계 8달러 설정):  
<https://platform.openai.com/settings/organization/billing/overview>
- 클로드(Claude) 가입(무료): <https://claude.ai/>
- Github 가입(무료): <https://github.com/>
- Github Copilot 가입(유료. 10달러/월): <https://github.com/features/copilot/plans>
- Huggingface 가입(무료): <https://huggingface.co>
- Huggingface API Token 생성(무료): <https://huggingface.co/settings/tokens>
- Stable diffusion - Kling 가입(유료. \$6.99 per month):  
<https://app.klingai.com/global/membership/membership-plan>
- gemini-api: <https://aistudio.google.com/app/apikey> gemini사용위한 google ai 키
- serp-api: <https://serpapi.com/manage-api-key> 구글 검색
- travily: <https://app.travily.com/home> 웹 검색
- wandb: <https://docs.wandb.ai/quickstart/> 모델학습 및 파인튜닝 모니터링 로그 도구
- langsmith:  
[https://docs.smith.langchain.com/administration/how\\_to\\_guides/organization\\_management/create\\_account\\_api\\_key](https://docs.smith.langchain.com/administration/how_to_guides/organization_management/create_account_api_key) (<https://smith.langchain.com/o/2c462eb1-5b83-41c8-96c5-e008809d5655/settings>) langchain 로그 & 디버그 도구
- Figma: <https://www.figma.com>
- visily ai: <https://app.visily.ai> 기획 AI 도구

# 머신러닝 기초

## 분류와 회귀의 차이

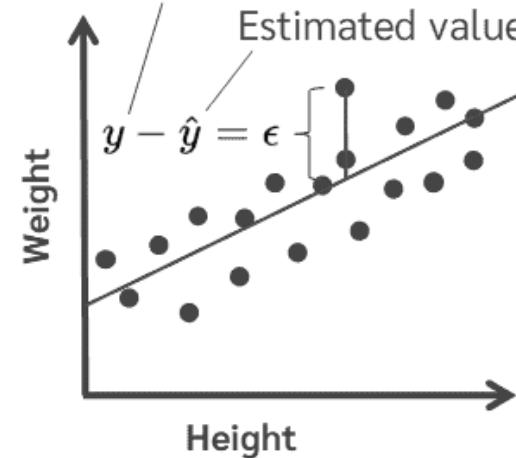
### Classification



분류는 입력 데이터를 미리 정의된 이산적인 범주(클래스) 중 하나로 할당하는 기계 학습

### Regression analysis

True value



회귀분석은 입력값과 대상값간의 관계를 최소오차가 되도록 학습. 입력에 대한 결과  $y$ 를 추정할 수 있는 모델

#### 특징

#### 분류 (Classification)

#### 목표

데이터를 사전 정의된 범주로 분류/할당

#### 출력 유형

이산적이고 범주형인 레이블 (예: "예/아니오", "A/B/C")

#### 문제 예시

스팸 감지, 질병 진단, 이미지 인식

#### 회귀분석 (Regression)

연속적인 숫자 값을 예측/추정

연속적이고 수치형인 값 (예: 1.23, 100,000)

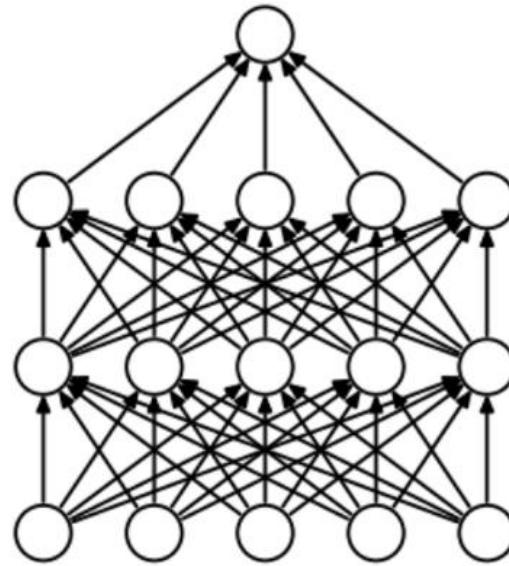
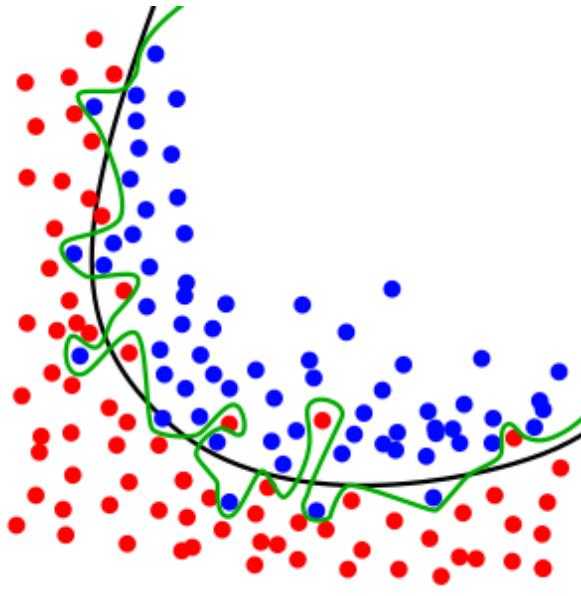
주택 가격 예측, 온도 예측, 판매량 예측

# 머신러닝 기초

Overfitting, 정규화, 검증셋 개념

## Overfitting

훈련 데이터에 너무 과하게 학습되어 실제 새로운 데이터에 대한 예측 성능이 저하되는 현상



# 머신러닝 기초

## Overfitting, 정규화, 검증셋 개념

### Overfitting과 검증셋

Korean Society of Magnetic Resonance in Medicine, 2019,  
Deep Learning in MR Image Processing



#### Training set

to train  
the model

#### Validation set

to monitor and tune  
the model

#### Test set

to evaluate  
the accuracy of the model

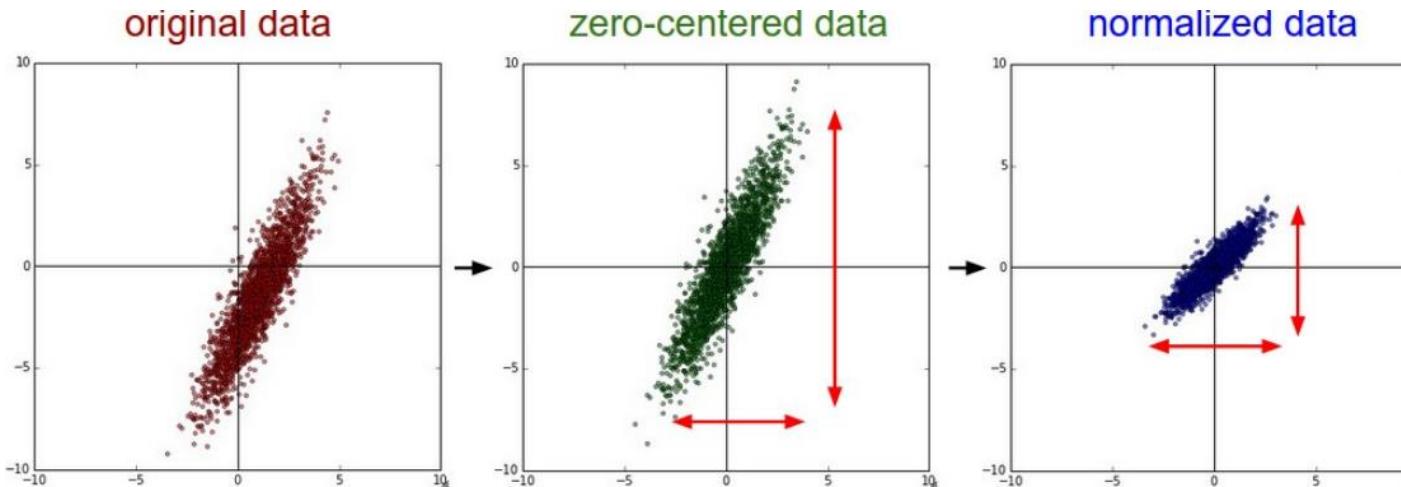
특징	훈련 세트 (Training Set)	검증 세트 (Validation Set)	테스트 세트 (Test Set)
목적	모델 학습 및 파라미터 최적화	하이퍼파라미터 튜닝 및 모델 선택	모델의 최종 성능 평가 및 일반화 능력 측정
사용 시기	모델 개발의 초기 및 반복적 학습 단계	학습 중간 및 하이퍼파라미터 조정 단계	모델 개발 완료 후 단 한 번만 사용
데이터 노출	모델이 직접 학습하는 데이터	모델이 학습 중 성능 평가에 사용되지만, 직접 학습에 사용되지는 않음	모델이 전혀 보지 못한 완전히 새로운 데이터
과적합 방지	훈련 데이터에 과적합될 수 있음	과적합 여부를 모니터링하고 방지 하는 데 도움	모델의 진정한 일반화 성능을 객관적으로 측정

# 머신러닝 기초

Overfitting, 정규화, 검증셋 개념

## Normalization

정규화(Regularization)는 머신러닝 모델이 훈련 데이터에 과적합(Overfitting)되는 것을 방지하고, 새로운 데이터에 대한 일반화 성능을 향상시키기 위한 기법들을 총칭



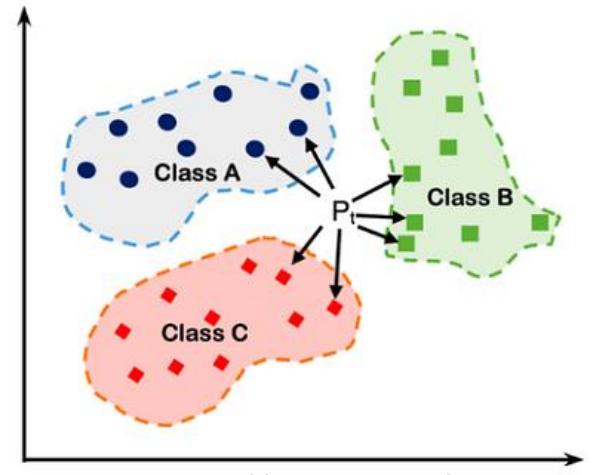
Practical Aspects  
of Deep Learning -  
2 | Ashish Jaiswal

기법	원리	주요 효과
L1 정규화	특정 가중치에 편중되는 현상 방지	과적합 완화
L2 정규화		모델 안정화, 과적합 완화
드롭아웃	학습 중 임의로 일부 뉴런 출력을 0으로 설정	뉴런 간 과도한 의존 억제, 일반화 성능 향상
조기 종료	검증용 데이터 성능 개선 없으면 학습을 중단	과적합 방지, 학습 시간 절약
데이터 증강	회전·이동·뒤집기·잡음 추가 등 변형을 통해 자료 증강	데이터 다양성 확대, 변형에 강한 모델, 일반화 성능 향상

# 머신러닝 기초 주요 알고리즘

## K-Nearest Neighbors (KNN)

이웃 기반 학습 (Instance-based learning)으로, 새 데이터 포인트가 주어졌을 때, 주변의 가장 가까운 K개의 훈련 데이터 포인트를 찾아 클러스터링 수행함. 데이터 분류에 사용



[K Nearest Neighbours — Introduction to Machine Learning Algorithms | by Sachinoni](#)

Function KNN\_Predict(training\_data, training\_labels, new\_data\_point, K):

```
distance_list = empty list
```

```
# Calculate distance to all training data points
```

```
For each training_data_point in training_data:
```

```
    distance = Calculate_Distance(new_data_point, training_data_point) # Euclidean Distance
```

```
    Add (distance, training_data_point_label) pair to distance_list
```

```
# Sort by distance and select the K nearest
```

```
Sort distance_list in ascending order by distance
```

```
k_nearest_neighbors = top K items from distance_list
```

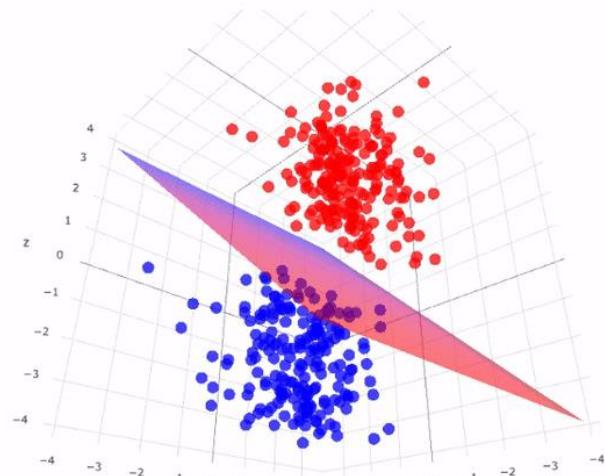
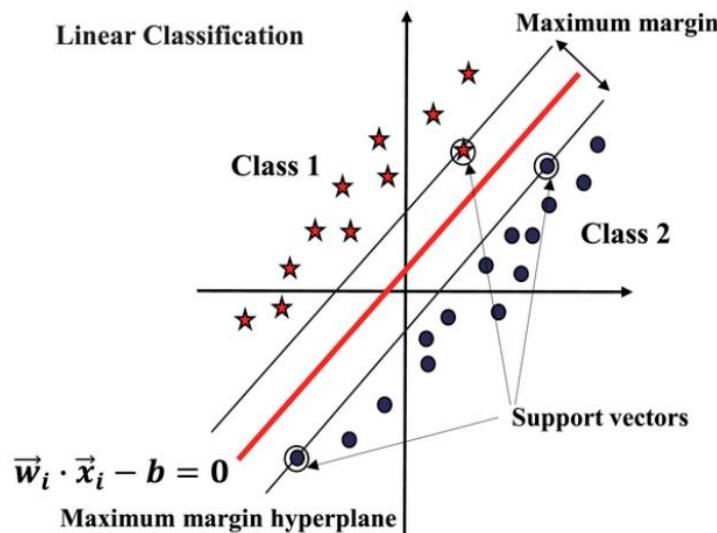
```
# Make Prediction
```

```
Return the most frequent label among k_nearest_neighbors_labels
```

# 머신러닝 기초 주요 알고리즘

## Support Vector Machine (SVM)

주어진 데이터를 서로 다른 클래스로 나누는 최적의 결정 경계(Decision Boundary)를 계산. 이를 통해 데이터셋을 분류함



# 머신러닝 기초 주요 알고리즘

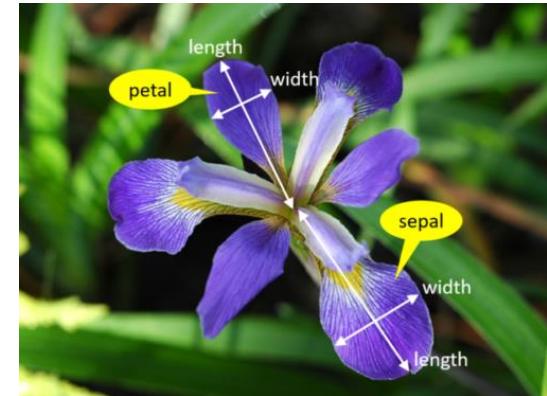
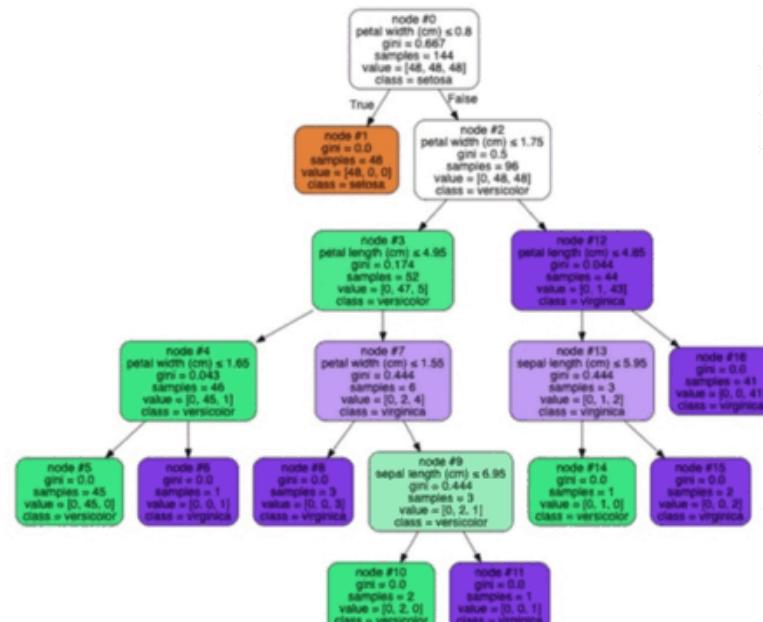
## Decision Tree

데이터를 특정 기준에 따라 연속적으로 분할하는 일련의 규칙을 나무(Tree) 구조로 표현하는 알고리즘

루트 노드 (Root Node): 트리의 시작점이며, 전체 데이터를 포함  
내부 노드 (Internal Node): 특정 특성(Feature)에 대한 '질문' 또는 '조건'을 나타냄. 질문 결과에 따라 데이터가 하위 노드로 분할  
리프 노드 (Leaf Node): 트리 끝점으로, 더 이상 분할되지 않으며 최종적인 결정(클래스 레이블 또는 예측 값)을 포함

### TEST DATA

### DECISION TREE



Dataset Order	Sepal length	Sepal width
1	5.1	3.5
2	4.9	3.0
101	6.3	3.3
102	5.8	2.7

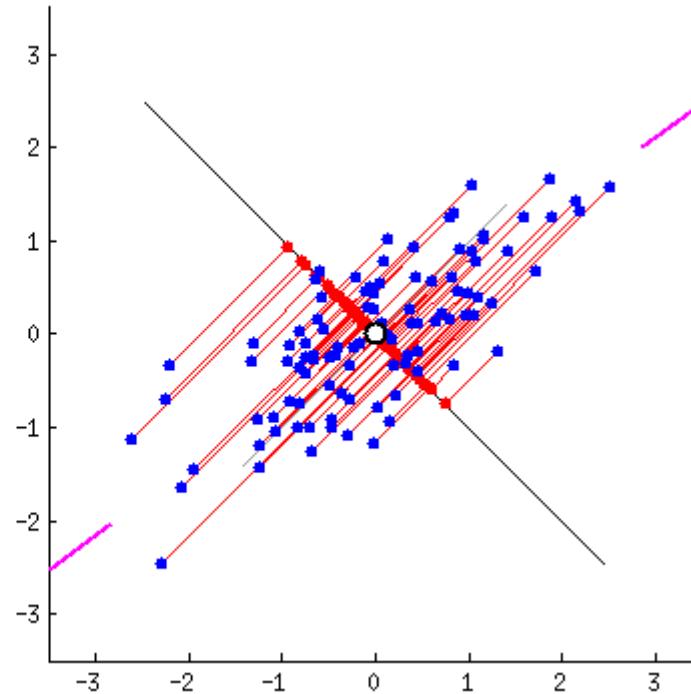
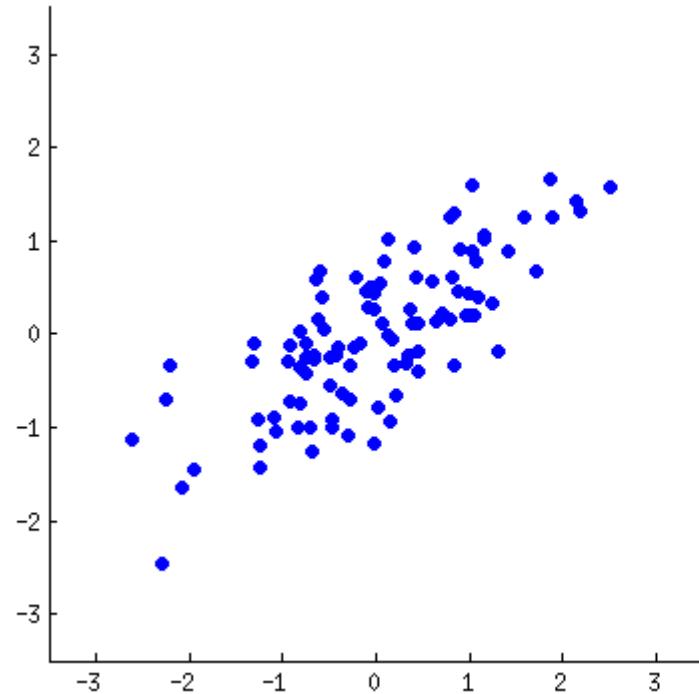
Petal length	Petal width	Species
1.4	0.2	I. setosa
1.4	0.2	I. setosa
6.0	2.5	I. virginica
5.1	1.9	I. virginica

# 머신러닝 기초

## 주요 알고리즘

### Principal Component Analysis (PCA)

고차원의 데이터를 저차원 공간으로 변환, 원본 데이터의 분산을 가장 잘 설명하는 직교 축을 계산. 데이터 차원 축소에 사용



# 머신러닝 기초

## 주요 알고리즘

알고리즘	구현 원리	응용
K-Nearest Neighbors (KNN)	새 데이터에 가장 가까운 K개 이웃 값 을 예측에 사용	추천 시스템, 이미지 분류 등.
Linear Regression	입력과 타겟 간의 최적 선형 관계를 찾아 예측함	가격 예측, 통계 추론 등.
Logistic Regression	선형 조합에 시그모이드를 적용해 확률 을 출력하고 이진 분류함.	스팸 분류, 질병 진단 등.
Support Vector Machine (SVM)	데이터를 나누는 최대 마진을 갖는 최 적의 면 경계를 계산	이미지 분류, 필기체 인식 등.
Decision Tree	데이터를 규칙에 따라 트리 구조로 분 할하여 예측함	의사결정 규칙 생성, 고객 분류 등.
Random Forest	여러 결정 트리를 생성하여 예측을 종 합하는 앙상블 학습	사기 탐지, 의료 진단 등.
Principal Component Analysis (PCA)	고차원 데이터를 분산 보존하며 저차원 으로 축소함.	이미지 압축, 특성 추출 등.

# 머신러닝 기초

Scikit-learn을 활용한 파이프라인 예시

## Scikit-learn

오픈 소스 머신러닝 라이브러리. 지도 학습(분류, 회귀), 비지도 학습(클러스터링, 차원 축소) 알고리즘, 데이터 전처리, 모델 평가 도구 제공

알고리즘: 선형 회귀, 로지스틱 회귀, 결정 트리, 랜덤 포레스트, K-NN, K-Means 등 다수의 알고리즘을 제공합니다.

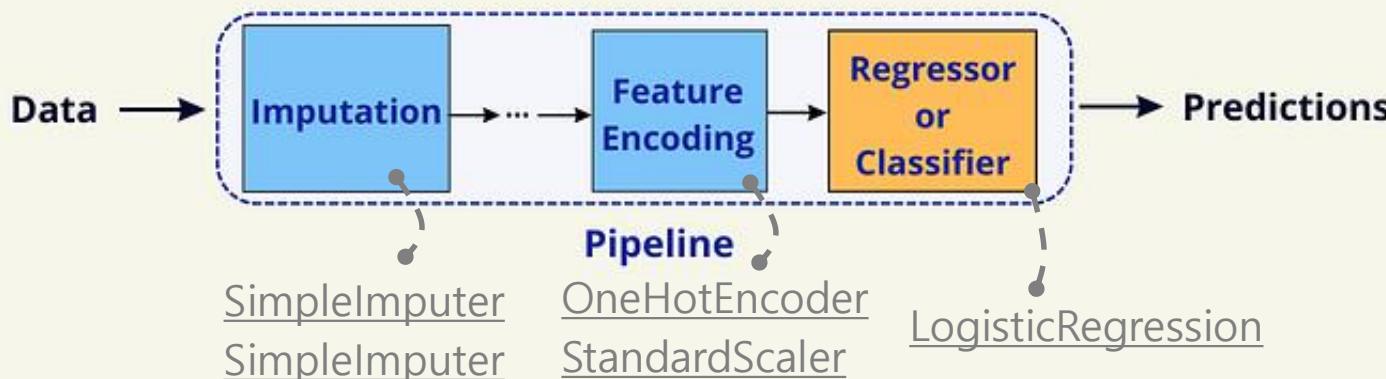
데이터 전처리: 스케일링(StandardScaler, MinMaxScaler), 인코딩(OneHotEncoder), 특성 변환(PolynomialFeatures) 등 데이터 정제 및 변환 도구

모델 선택 및 평가: 교차 검증(cross\_val\_score), 하이퍼파라미터 튜닝(GridSearchCV, RandomizedSearchCV), 성능 지표(accuracy, classification) 등 제공합니다.

일관된 API: 모든 추정기(Estimator)가 fit(), transform(), predict()와 같은 일관된 메서드를 사용합니다.

Pipeline은 여러 변환기(transformer)와 하나의 최종 추정기(estimator)를 순차적으로 연결하여 하나의 결합된 추정기처럼 작동하게 하는 클래스입니다.

## Simplify Machine Learning Workflow With Scikit-Learn Pipelines



# 머신러닝 기초

Scikit-learn을 활용한 파이프라인 예시

```
import numpy as np, pandas as pd
from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline, make_pipeline
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.model_selection import train_test_split
```

```
data = {
    "state": ["CA", "WA", "CA", np.nan, "NV", "WA"],
    "gender": ["male", "female", "female", "male", np.nan, "female"],
    "age": [34, 29, 22, 44, 55, np.nan],
    "weight": [122, 150, 130, np.nan, 140, 175],
    "target": [0, 1, 0, 1, 0, 1]
}
df = pd.DataFrame(data)
```

```
X = df.drop("target", axis=1)
print("Input DataFrame:")
print(X)
```

```
y = df["target"]
print("Target Series:")
print(y)
```

	state	gender	age	weight	
0	CA	male	34.0	122.0	0
1	WA	female	29.0	150.0	1
2	CA	female	22.0	130.0	0
3	NaN	male	44.0	NaN	1
4	NV	NaN	55.0	140.0	0
5	WA	female	NaN	175.0	1

# 머신러닝 기초

Scikit-learn을 활용한 파이프라인 예시

```
categorical_preprocessor = Pipeline(  
    steps=[ ("imputation_constant", SimpleImputer(fill_value="missing", strategy="constant")),  
           ("onehot", OneHotEncoder(handle_unknown="ignore")) ]  
)
```

CA:	[1, 0, 0, 0]
WA:	[0, 1, 0, 0]
NV:	[0, 0, 1, 0]
missing:	[0, 0, 0, 1]

```
numeric_preprocessor = Pipeline(  
    steps=[ ("imputation_mean", SimpleImputer(missing_values=np.nan, strategy="mean")),  
           ("scaler", StandardScaler()) ]  
)
```

```
preprocessor = ColumnTransformer(  
    [  
        ("categorical", categorical_preprocessor, ["state", "gender"]),  
        ("numerical", numeric_preprocessor, ["age", "weight"]),  
    ]  
)
```

	state	gender	age	weight	
0	CA	male	34.0	122.0	0
1	WA	female	29.0	150.0	1
2	CA	female	22.0	130.0	0
3	NaN	male	44.0	NaN	1
4	NV	NaN	55.0	140.0	0
5	WA	female	NaN	175.0	1

```
pipe = make_pipeline(preprocessor, LogisticRegression(max_iter=500))  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)  
pipe.fit(X_train, y_train)
```

```
print(f"Input test set: {X_test}")  
predictions = pipe.predict(X_test)  
print(f"Predictions on the test set: {predictions}")
```

	state	gender	age	weight
0	CA	male	34.0	122.0
1	WA	female	29.0	150.0
Predictions on the test set:				
[0 1]				

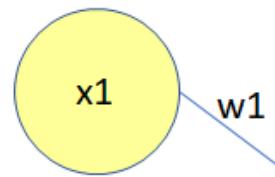
# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)  
Optimizer, Learning Rate, Batch Size 의미  
CNN/RNN/LSTM의 개념적 차이  
Tensorflow vs Pytorch 비교

# 딥러닝의 구조적 이해

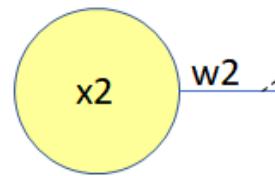
신경망 기본 구조 (Perceptron, MLP)

Input  
independent  
variable 1



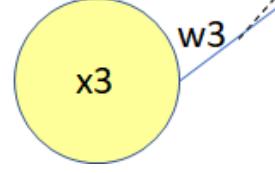
$w_1$

Input  
independent  
variable 2



$w_2$

Input  
independent  
variable 3



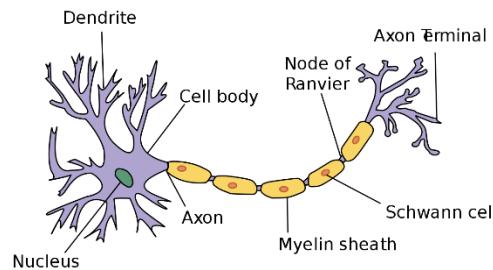
$w_3$

synapses

Neuron

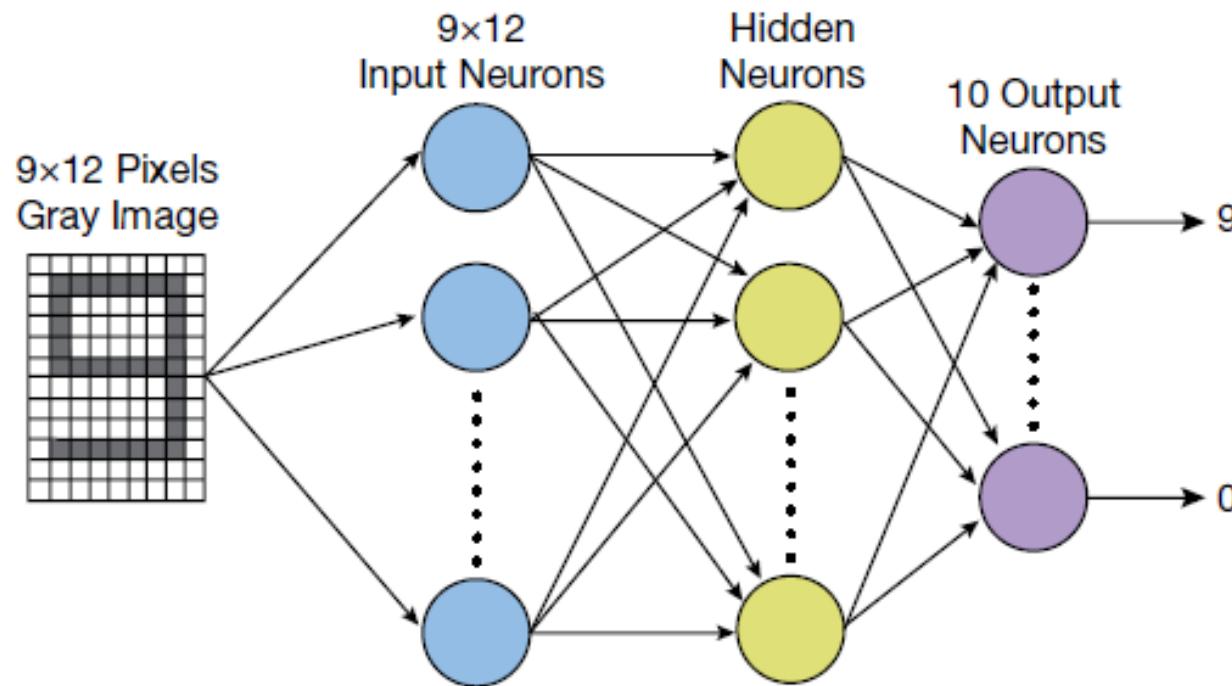
$y$

Output  
dependent  
variable



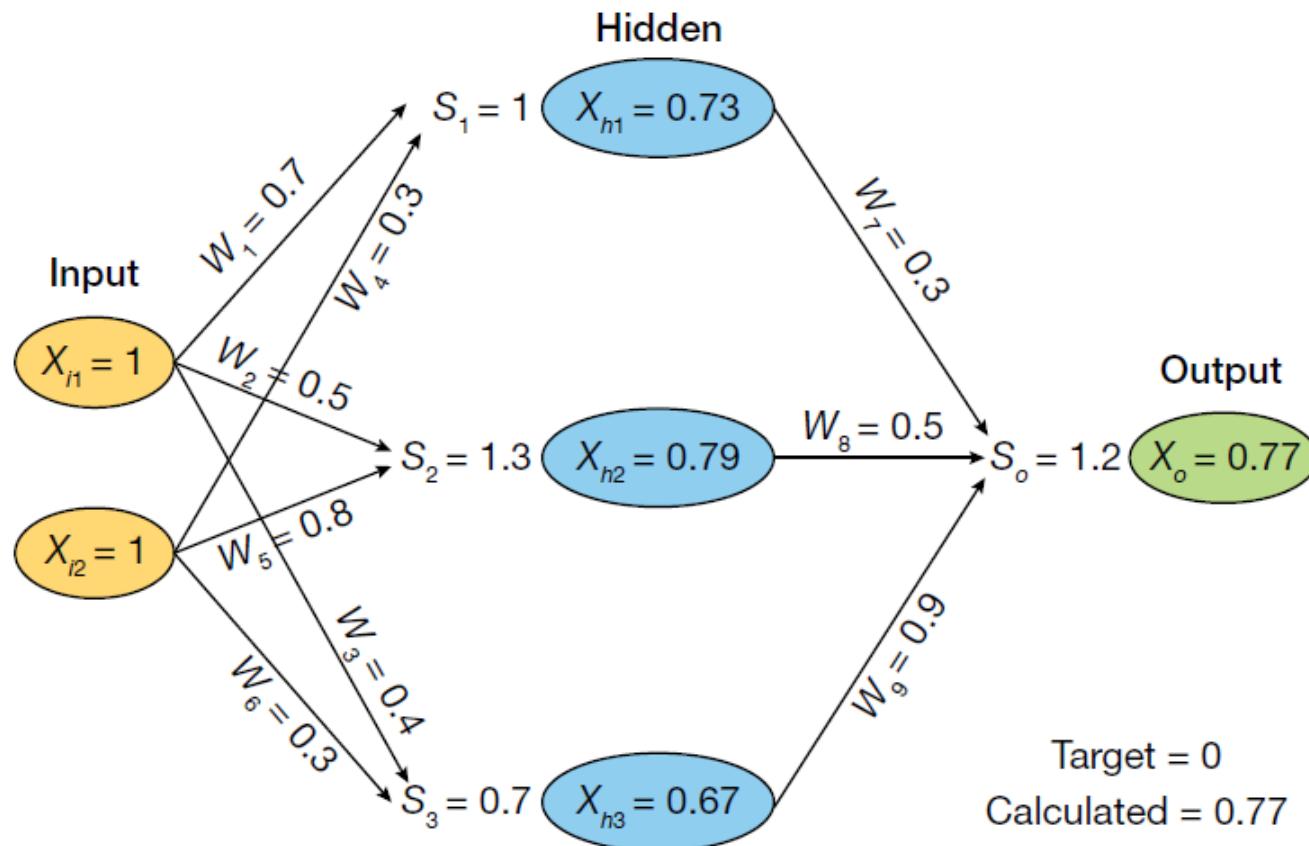
# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



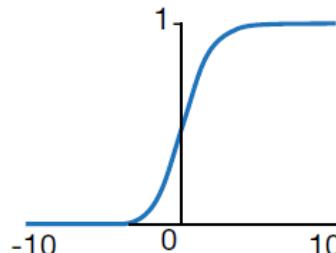
# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



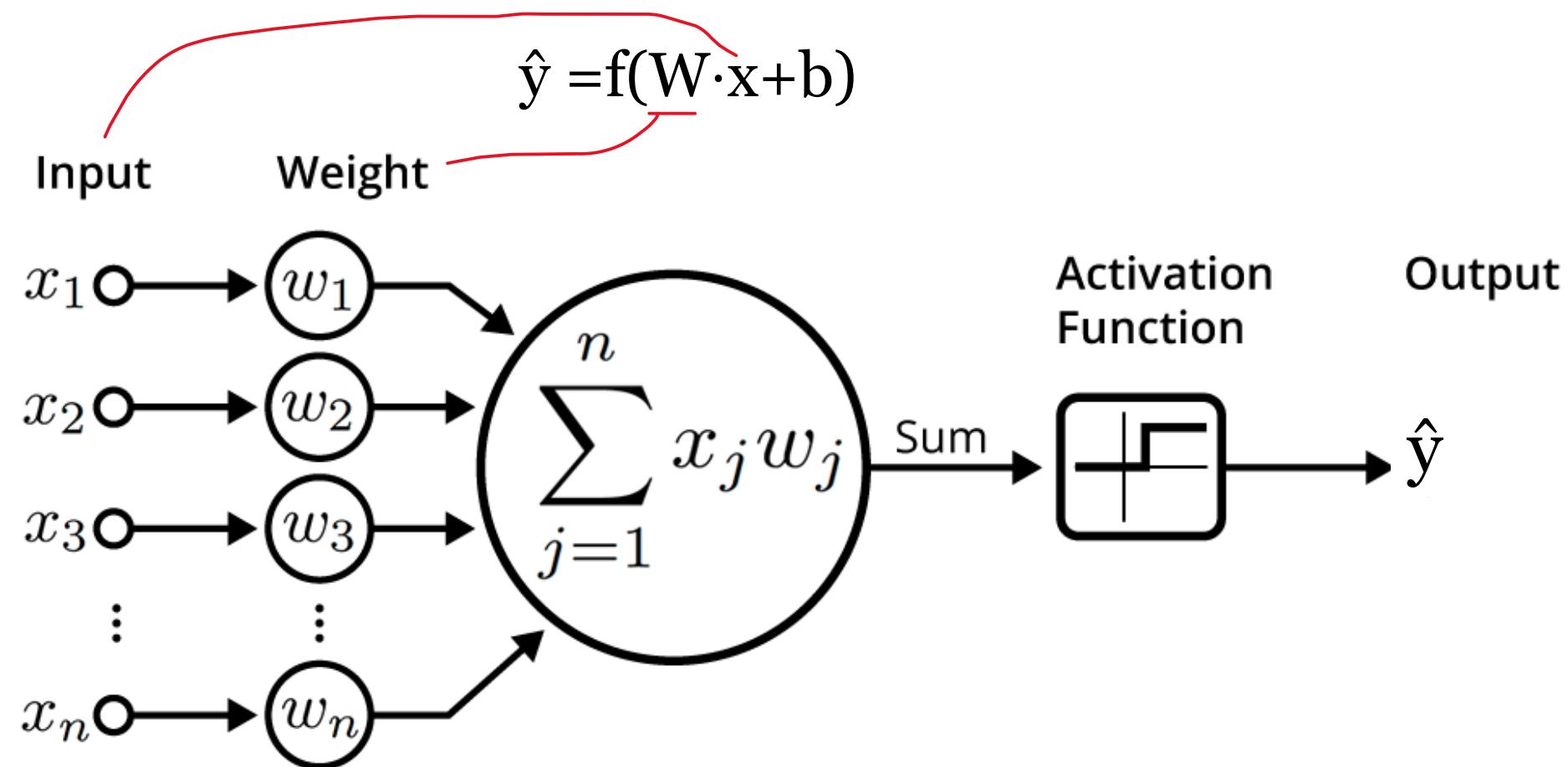
Sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

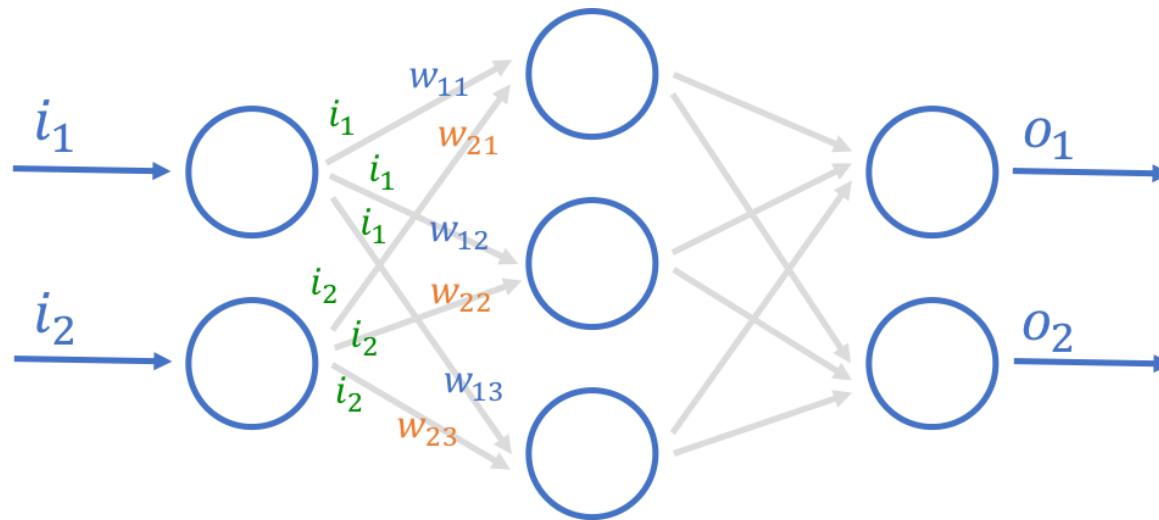


$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2.$$

Mean  
Squared  
Error

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

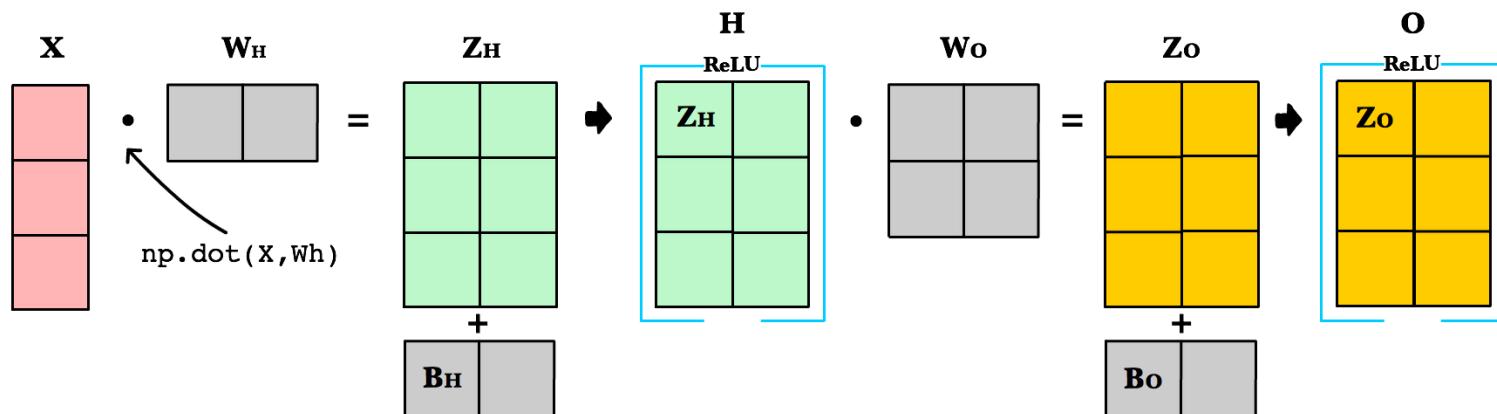
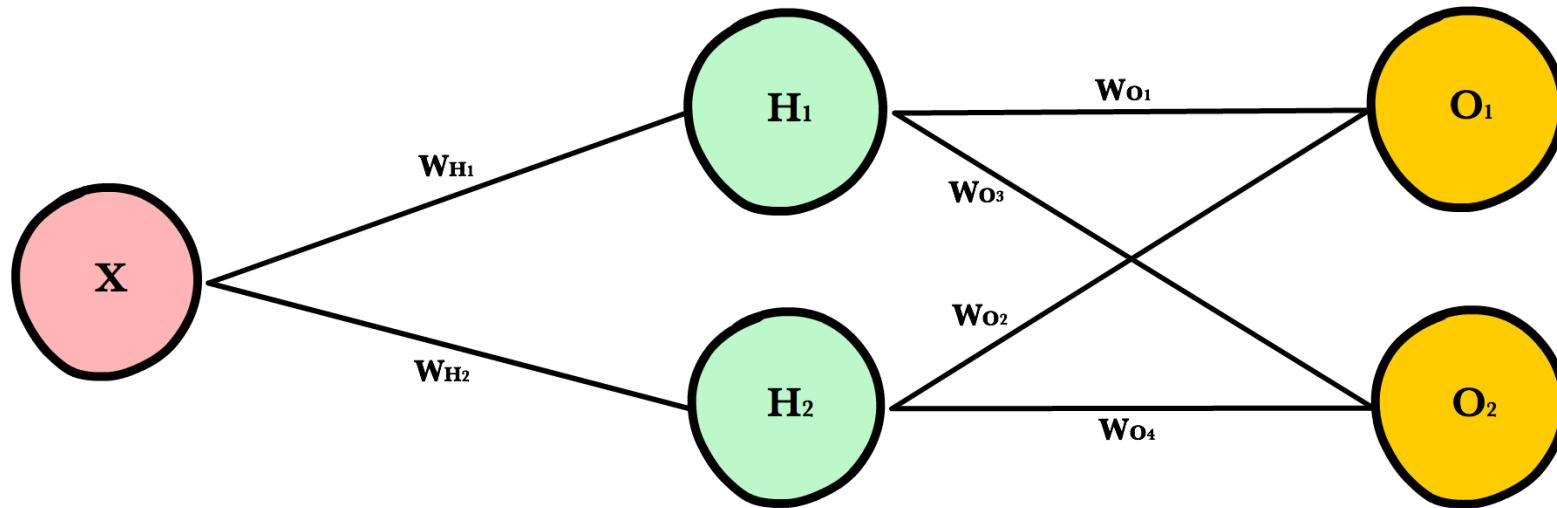


$$f(W \cdot x + b) = \hat{y}$$

$$\begin{bmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \\ w_{13} & w_{23} \end{bmatrix} \cdot \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} (w_{11} \times i_1) + (w_{21} \times i_2) \\ (w_{12} \times i_1) + (w_{22} \times i_2) \\ (w_{13} \times i_1) + (w_{23} \times i_2) \end{bmatrix}$$

# 딥러닝의 구조적 이해

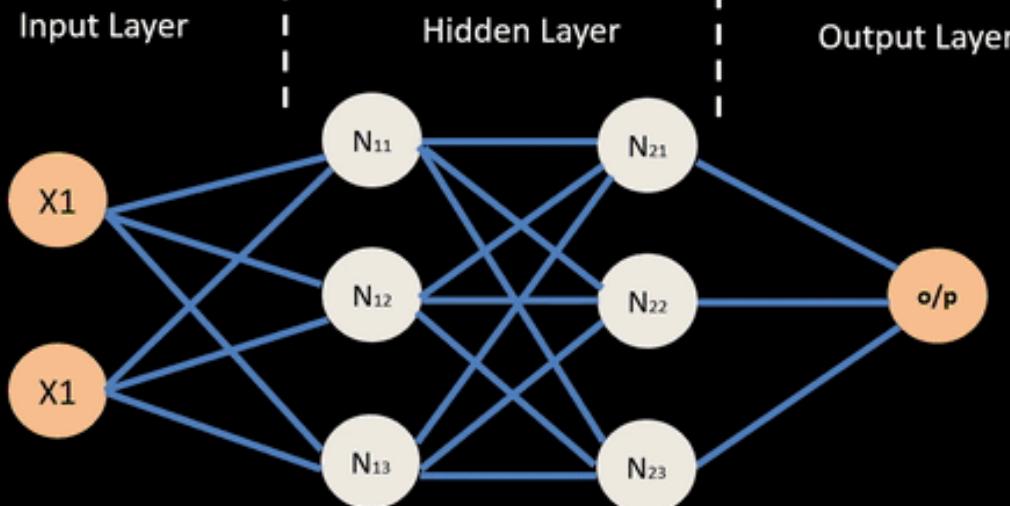
신경망 기본 구조 (Perceptron, MLP)



# 딥러닝의 구조적 이해

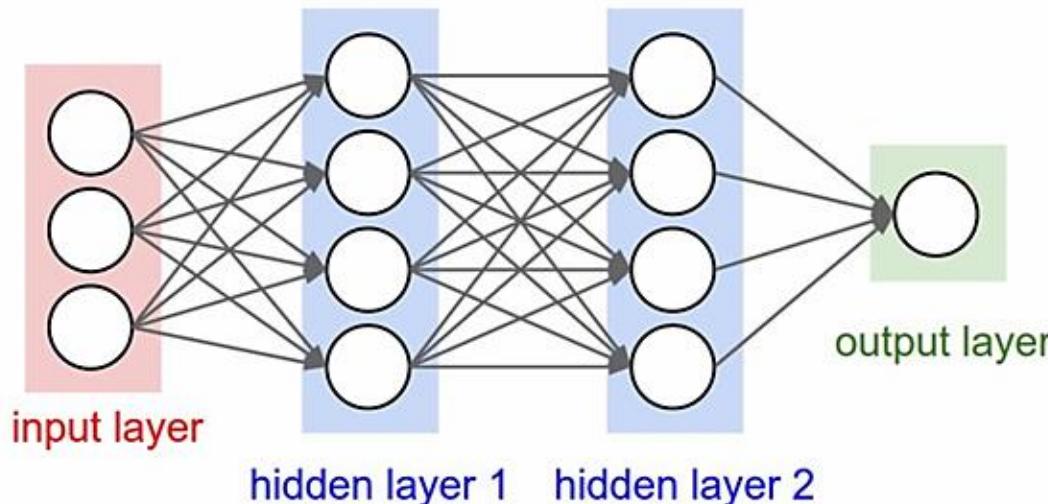
신경망 기본 구조 (Perceptron, MLP)

## Neural Network – Backpropagation



# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



```
# forward-pass of a 3-layer neural network:  
f = lambda x: 1.0/(1.0 + np.exp(-x)) # activation function (use sigmoid)  
x = np.random.randn(3, 1) # random input vector of three numbers (3x1)  
h1 = f(np.dot(W1, x) + b1) # calculate first hidden layer activations (4x1)  
h2 = f(np.dot(W2, h1) + b2) # calculate second hidden layer activations (4x1)  
out = np.dot(W3, h2) + b3 # output neuron (1x1)
```

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

## 역전파(Backpropagation)

인공 신경망 가중치(weights)를 효율적으로 업데이트하여 모델을 훈련시키는 알고리즘.

연쇄 법칙(Chain Rule)을 사용, 손실 함수 기울기(gradient)를 계산, 이 기울기를 사용하여 경사 하강법(Gradient Descent)과 같은 알고리즘으로 가중치 업데이트

$$\Delta w_i = -\alpha \frac{dE}{dw_i}$$

$$w_{next} = w + \Delta w$$

$$\frac{\partial E_p}{\partial w_{ji}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial w_{ji}}$$

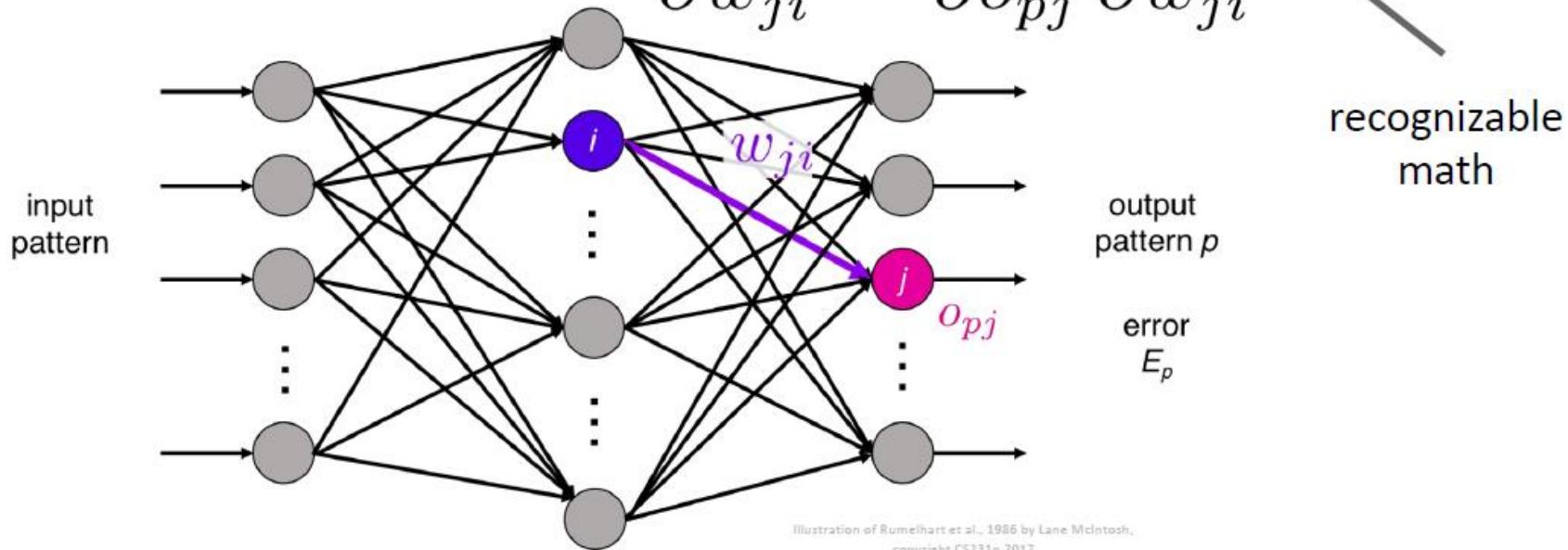


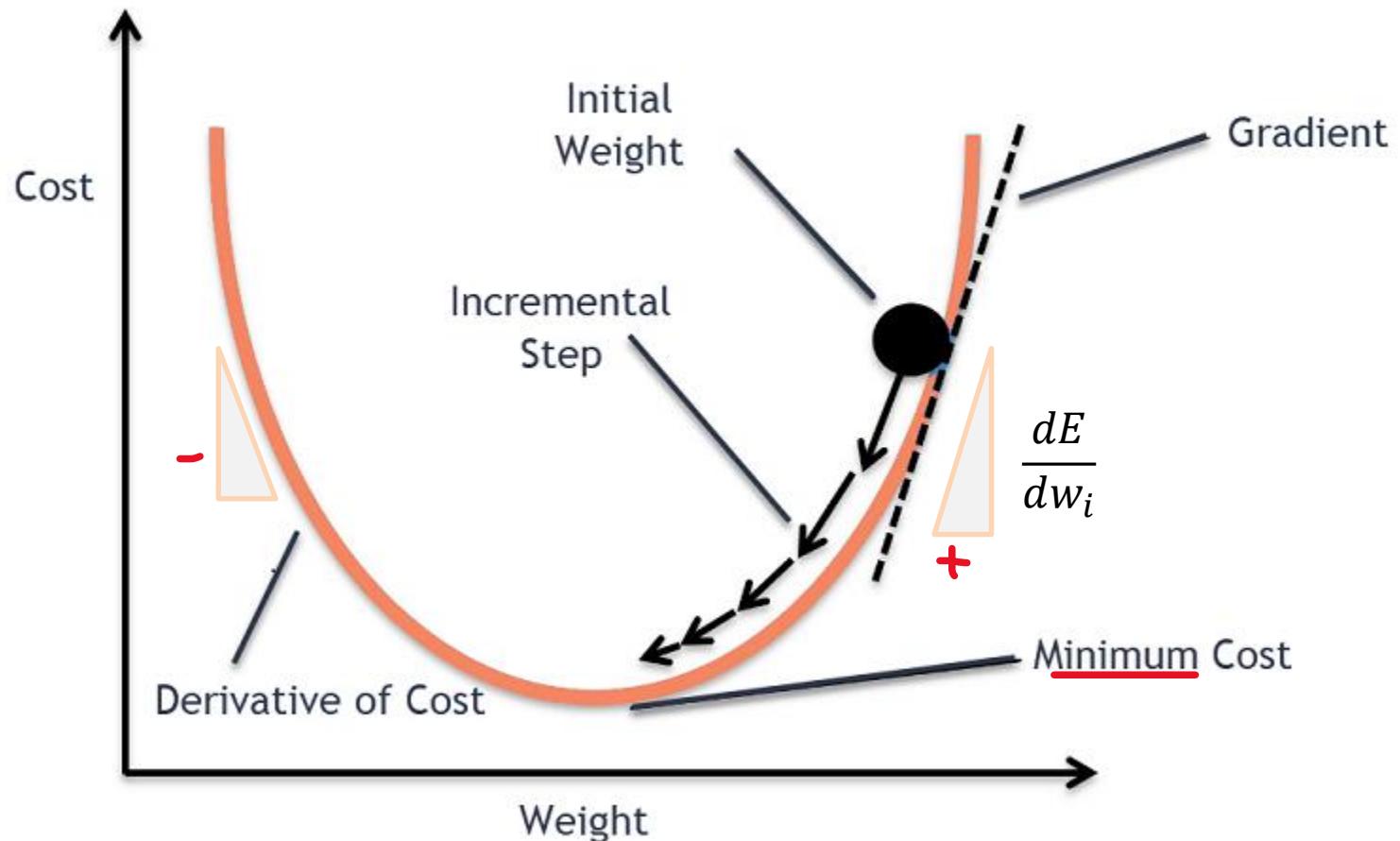
Illustration of Rumelhart et al., 1986 by Lane McIntosh,  
copyright CS231n 2017

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

$$\Delta w_i = -\alpha \frac{dE}{dw_i}$$

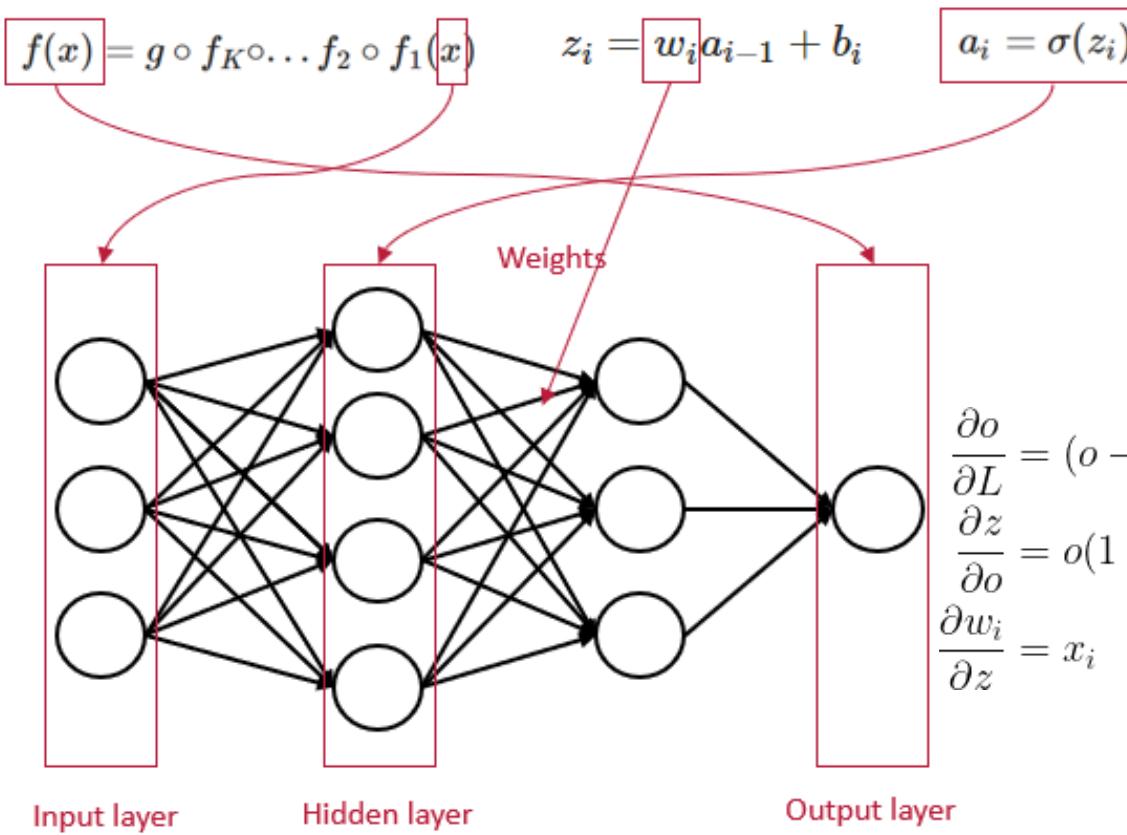
$$w_{next} = w + \Delta w$$



# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

신경망 계층은 단일 매개 변수의 단일 함수가 아닌, 여러 매개변수의 합성 함수  
가중치 미소량 변화에 따른 오차의 변화량 계산 위해 미적분학의 체인규칙(Chain Rule) 사용



$$\frac{\partial w_i}{\partial L} = \frac{\partial o}{\partial L} \times \frac{\partial z}{\partial o} \times \frac{\partial w_i}{\partial z}$$

$$\frac{\partial o}{\partial L} = (o - y) \quad (\text{derivative of the loss function})$$

$$\frac{\partial z}{\partial o} = o(1 - o) \quad (\text{derivative of the sigmoid function})$$

$$\frac{\partial w_i}{\partial z} = x_i \quad (\text{derivative of the weighted sum})$$

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

신경망 계층은 단일 매개 변수의 단일 함수가 아닌, 여러 매개변수의 합성 함수  
가중치 미소량 변화에 따른 오차의 변화량 계산 위해 미적분학의 체인규칙(Chain Rule) 사용

Rule	$f(x)$	Scalar derivative notation with respect to $x$	Example
Constant	$c$	0	$\frac{d}{dx} 99 = 0$
Multiplication by constant	$cf$	$c \frac{df}{dx}$	$\frac{d}{dx} 3x = 3$
Power Rule	$x^n$	$nx^{n-1}$	$\frac{d}{dx} x^3 = 3x^2$
Sum Rule	$f + g$	$\frac{df}{dx} + \frac{dg}{dx}$	$\frac{d}{dx} (x^2 + 3x) = 2x + 3$
Difference Rule	$f - g$	$\frac{df}{dx} - \frac{dg}{dx}$	$\frac{d}{dx} (x^2 - 3x) = 2x - 3$
Product Rule	$fg$	$f \frac{dg}{dx} + \frac{df}{dx}g$	$\frac{d}{dx} x^2 x = x^2 + x2x = 3x^2$
Chain Rule	$f(g(x))$	$\frac{df(u)}{du} \frac{du}{dx}$ , let $u = g(x)$	$\frac{d}{dx} \ln(x^2) = \frac{1}{x^2} 2x = \frac{2}{x}$

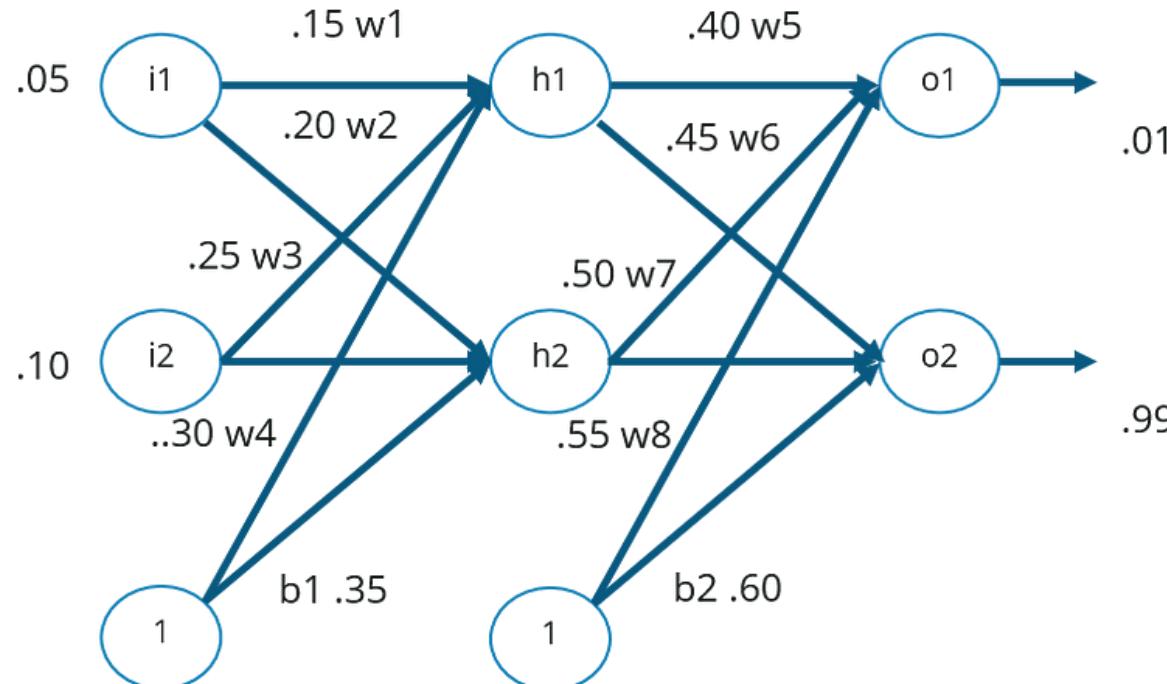
$$y=f(g(x))=\sin(x^2) \quad u=\underset{x}{\overset{\text{square}}{\underset{\uparrow}{}}} \left\{ \begin{array}{l} y=\sin \\ u=square \\ \frac{du}{dx} \end{array} \right\} \frac{dy}{du}$$

$$\frac{dy}{dx} = \frac{dy}{du} \frac{du}{dx} = \cos(u) 2x = 2x \cos(x^2)$$

$$u=x^2, y=\sin(u) \quad \frac{dy}{dx} = \frac{dy}{du} \frac{du}{dx} \quad \frac{du}{dx} = 2x \quad \frac{dy}{du} = \cos(u)$$

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



Net Input For  $h_1$ :

$$\text{net } h_1 = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$

$$\text{net } h_1 = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

Output Of  $h_1$ :

$$\text{out } h_1 = 1 / (1 + e^{-\text{net } h_1})$$

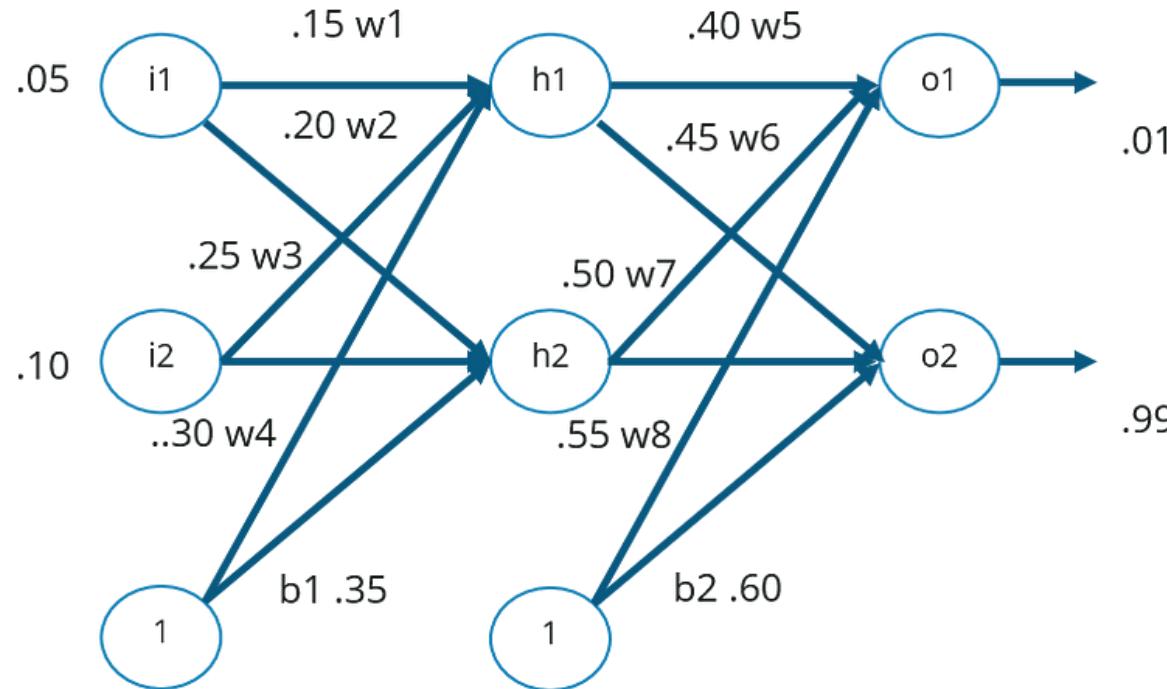
$$1 / (1 + e^{-0.3775}) = 0.593269992$$

Output Of  $h_2$ :

$$\text{out } h_2 = 0.596884378$$

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



Output For o1:

$$\text{net o1} = w5 * \text{out h1} + w6 * \text{out h2} + b2 * 1 \rightarrow 0.4 * 0.593269992 + 0.45 * 0.596884378 + 0.6 * 1 = 1.105905967$$

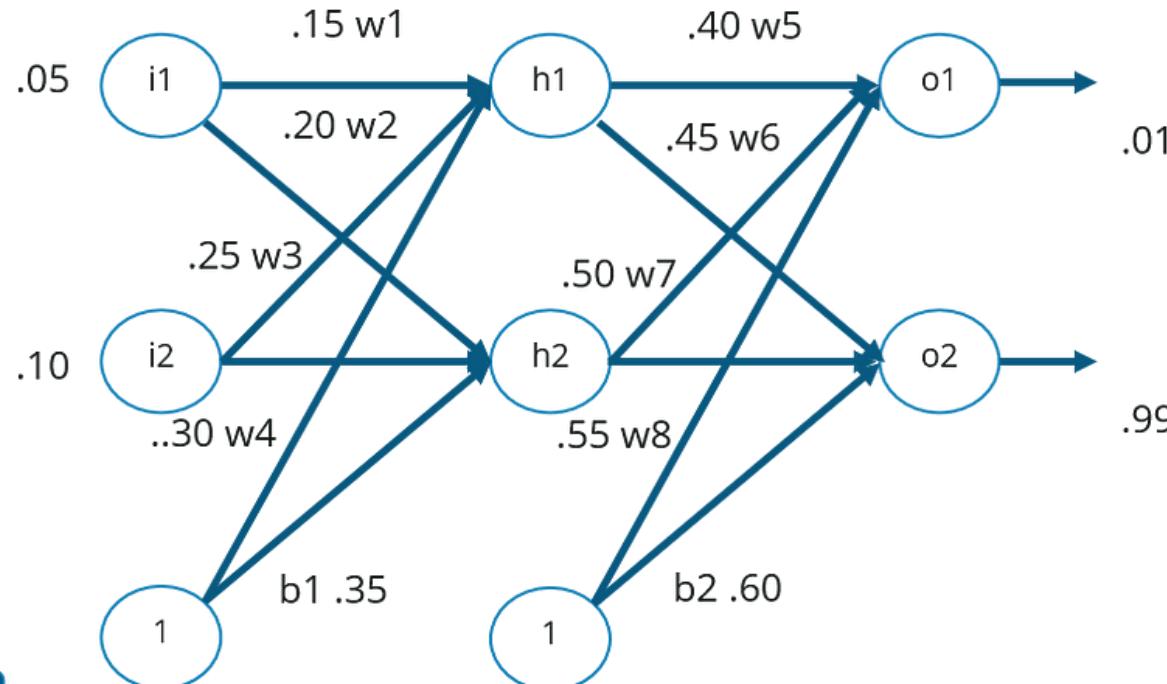
$$\text{Out o1} = 1 / (1 + e^{-\text{net o1}}) \rightarrow 1 / (1 + e^{-1.105905967}) = 0.75136507$$

Output For o2:

$$\text{Out o2} = 0.772928465$$

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



Error For o1:

$$E_{o1} = \frac{1}{2}(\text{target} - \text{output})^2$$

$$\frac{1}{2} (0.01 - 0.75136507)^2 = 0.274811083$$

Error For o2:

$$E_{o2} = 0.023560026$$

Total Error:

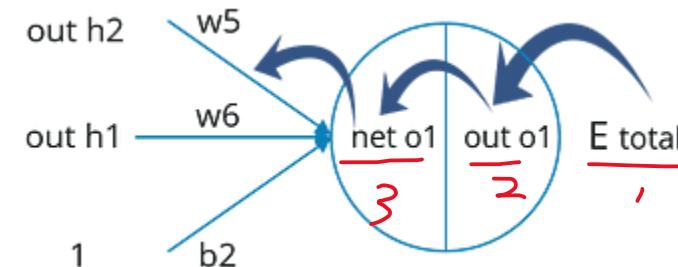
$$E_{\text{total}} = E_{o1} + E_{o2}$$

$$0.274811083 + 0.023560026 = 0.298371109$$

# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

$$\frac{\delta E_{total}}{\delta w_5} = \frac{\delta E_{total}}{\delta \text{out } o_1} * \frac{\delta \text{out } o_1}{\delta \text{net } o_1} * \frac{\delta \text{net } o_1}{\delta w_5}$$



$$E_{total} = 1/2(\text{target } o_1 - \text{out } o_1)^2 + 1/2(\text{target } o_2 - \text{out } o_2)^2$$

$$\frac{\delta E_{total}}{\delta \text{out } o_1} = -(\text{target } o_1 - \text{out } o_1) = -(0.01 - 0.75136507) = 0.74136507$$

$$\text{out } o_1 = 1/(1+e^{-\text{net } o_1})$$

$$\frac{\delta \text{out } o_1}{\delta \text{net } o_1} = \text{out } o_1 (1 - \text{out } o_1) = 0.75136507 (1 - 0.75136507) = 0.186815602$$

$$\text{net } o_1 = w_5 * \text{out } h_1 + w_6 * \text{out } h_2 + b_2 * 1$$

$$\frac{\delta \text{net } o_1}{\delta w_5} = 1 * \text{out } h_1 w_5^{(1-1)} + 0 + 0 = 0.593269992$$

$$\frac{\delta E_{total}}{\delta w_5} = \frac{\delta E_{total}}{\delta \text{out } o_1} * \frac{\delta \text{out } o_1}{\delta \text{net } o_1} * \frac{\delta \text{net } o_1}{\delta w_5} \quad 0.082167041$$

$$w_5^+ = w_5 - n \frac{\delta E_{total}}{\delta w_5} \quad w_5^+ = 0.4 - 0.5 * 0.082167041$$



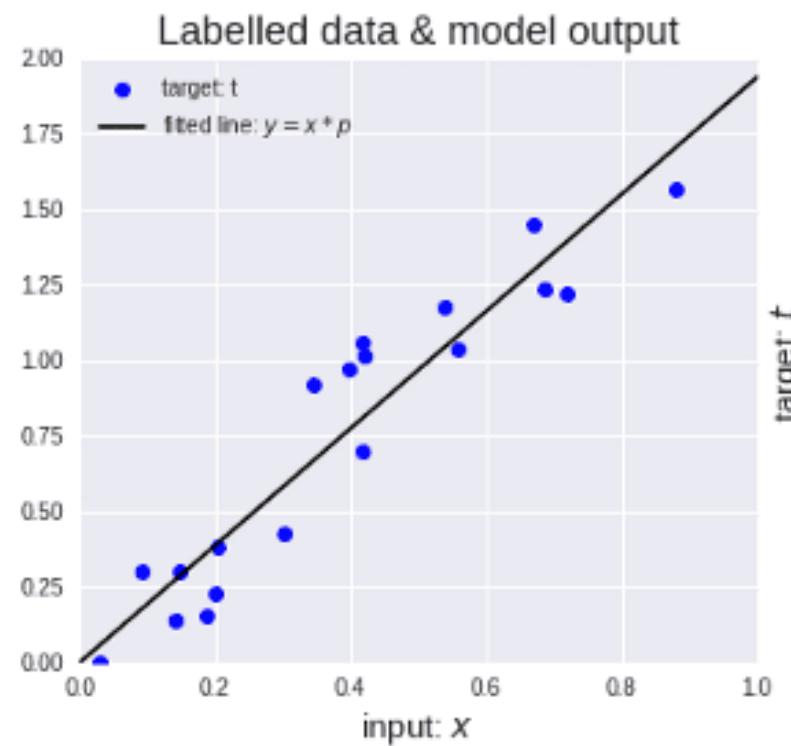
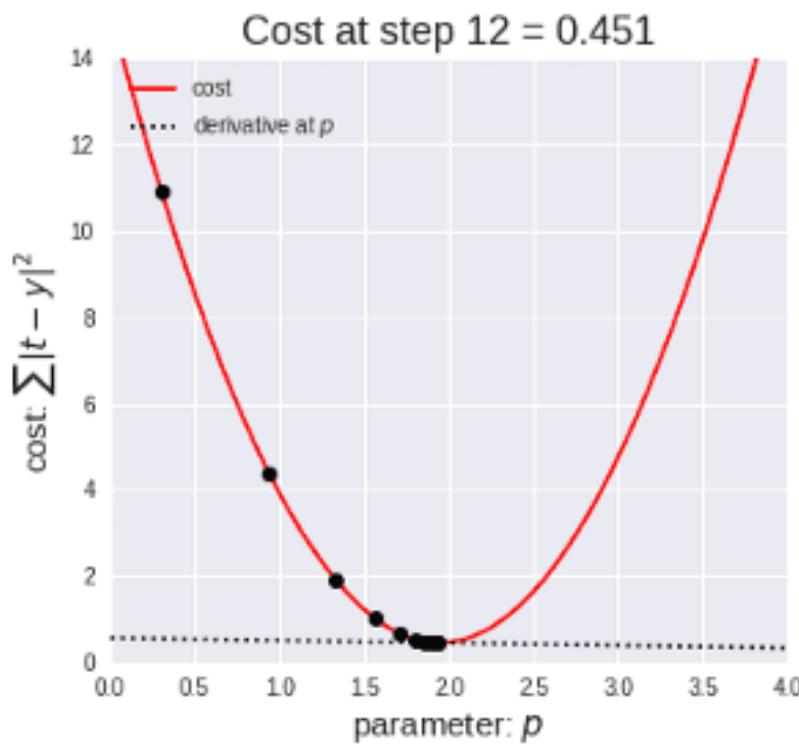
# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

$$\hat{y} = f(W \cdot x + b)$$

$$\text{target} = \min_{\text{LOSS}}(y - \hat{y})$$

$$w_{new} = w_{old} + \eta \Delta w$$

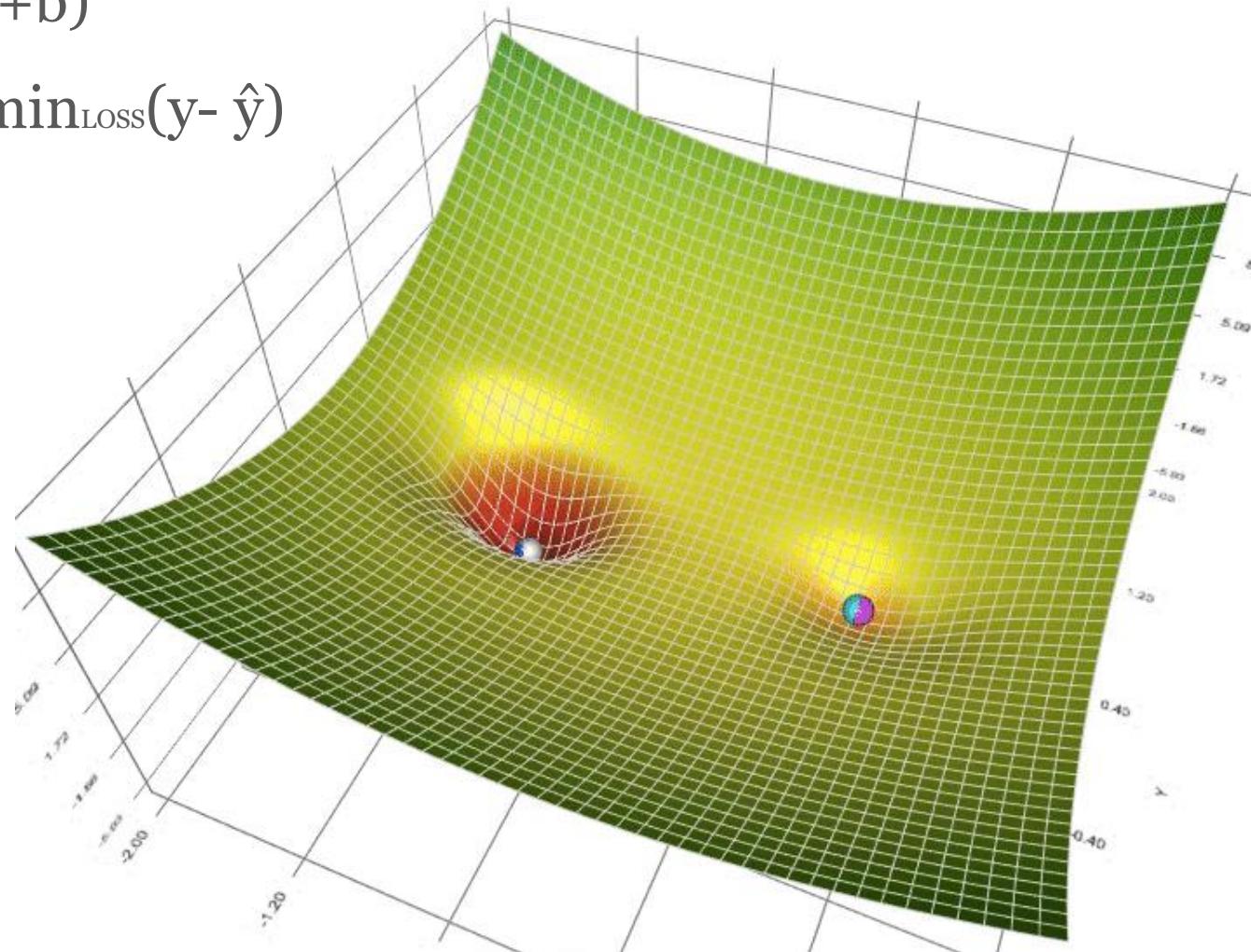


# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

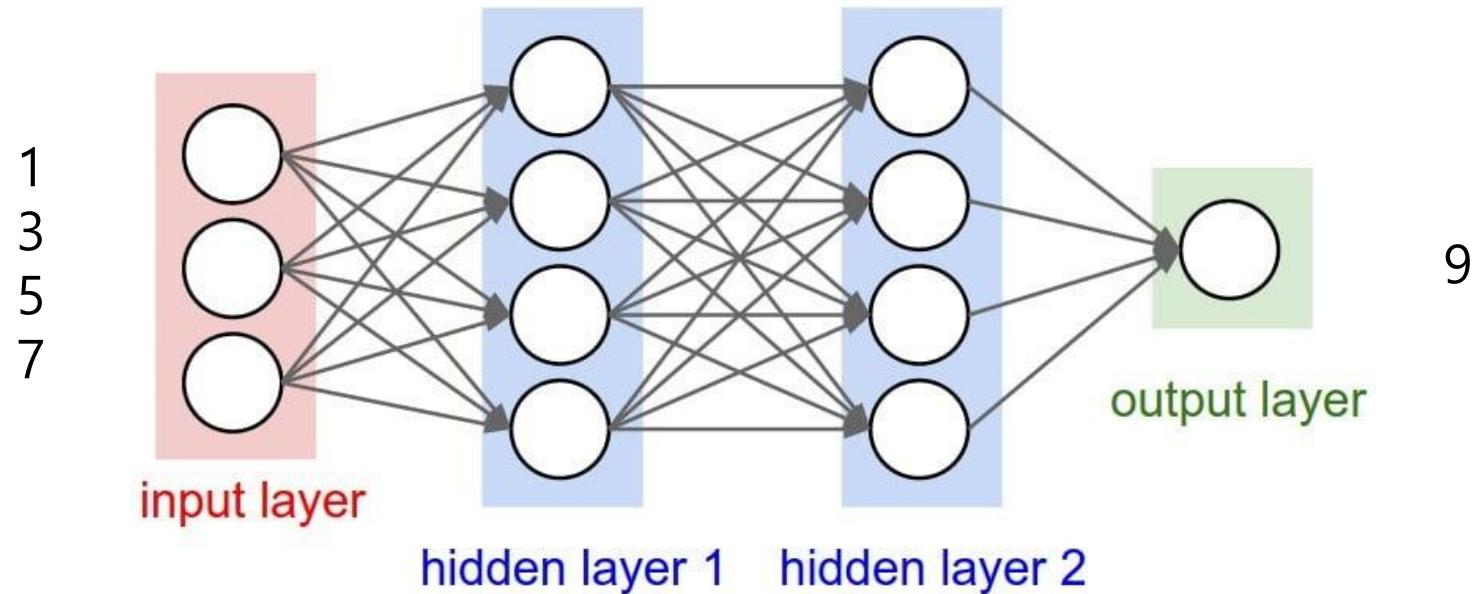
$$\hat{y} = f(W \cdot x + b)$$

$$\text{target} = \min_{\text{LOSS}}(y - \hat{y})$$



# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

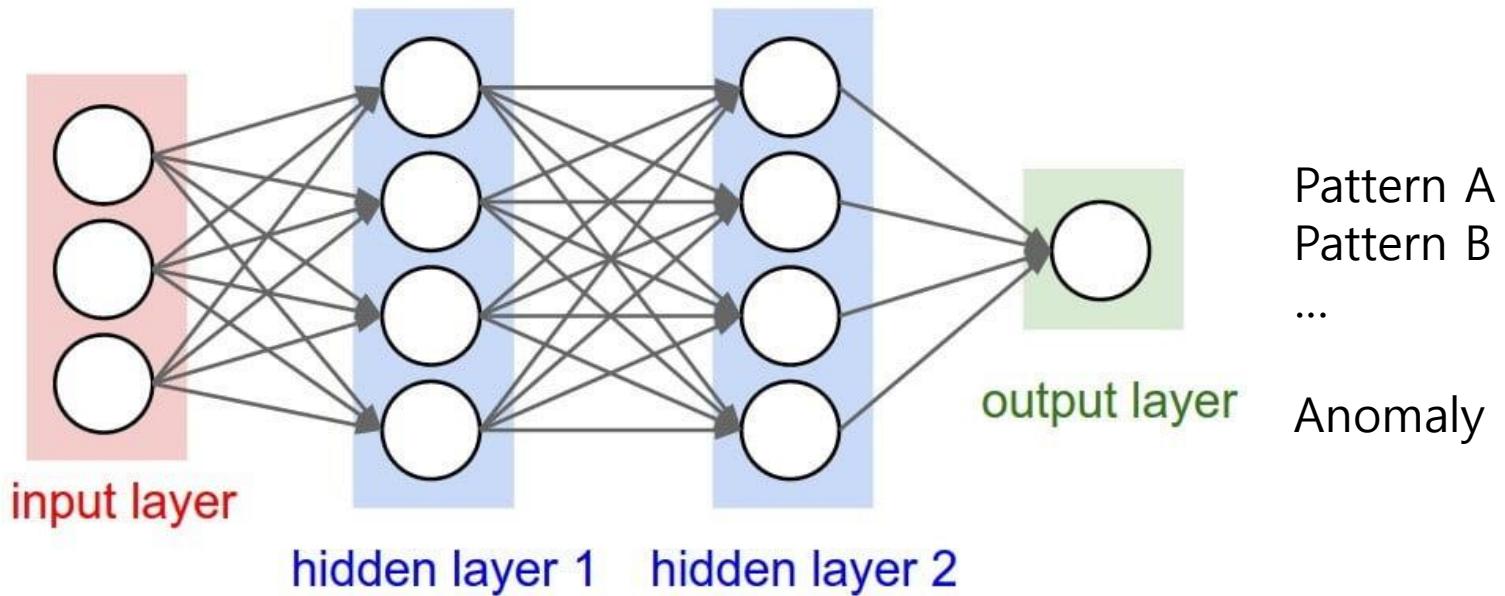


# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

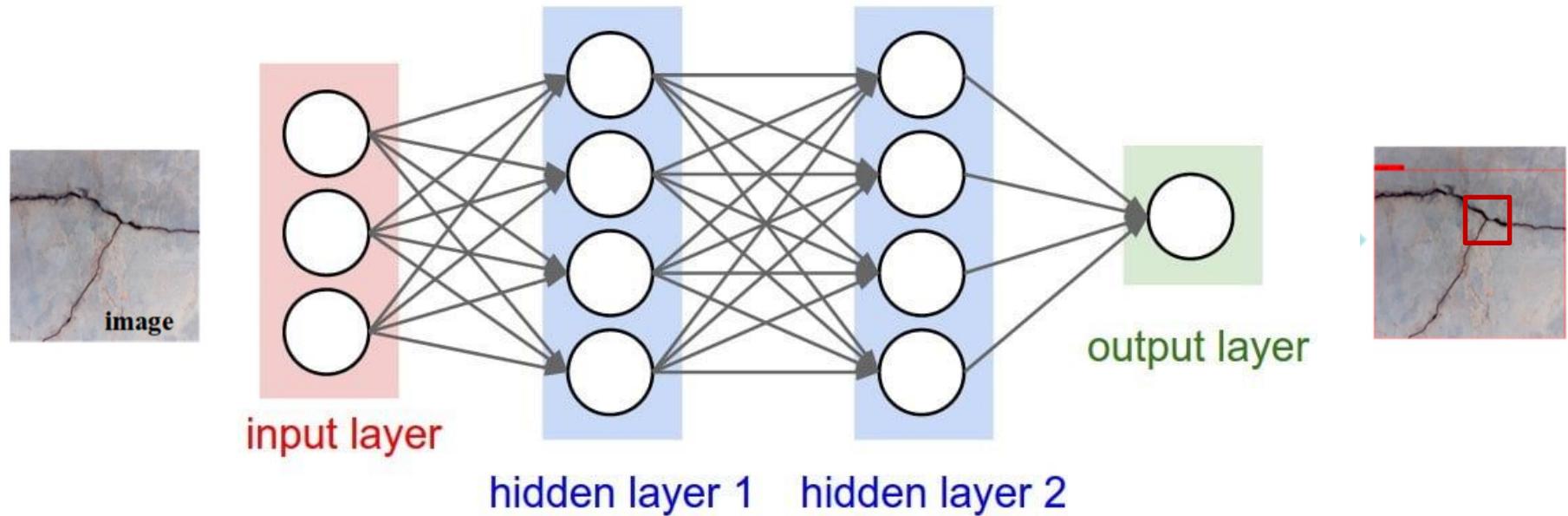


Sensor Data



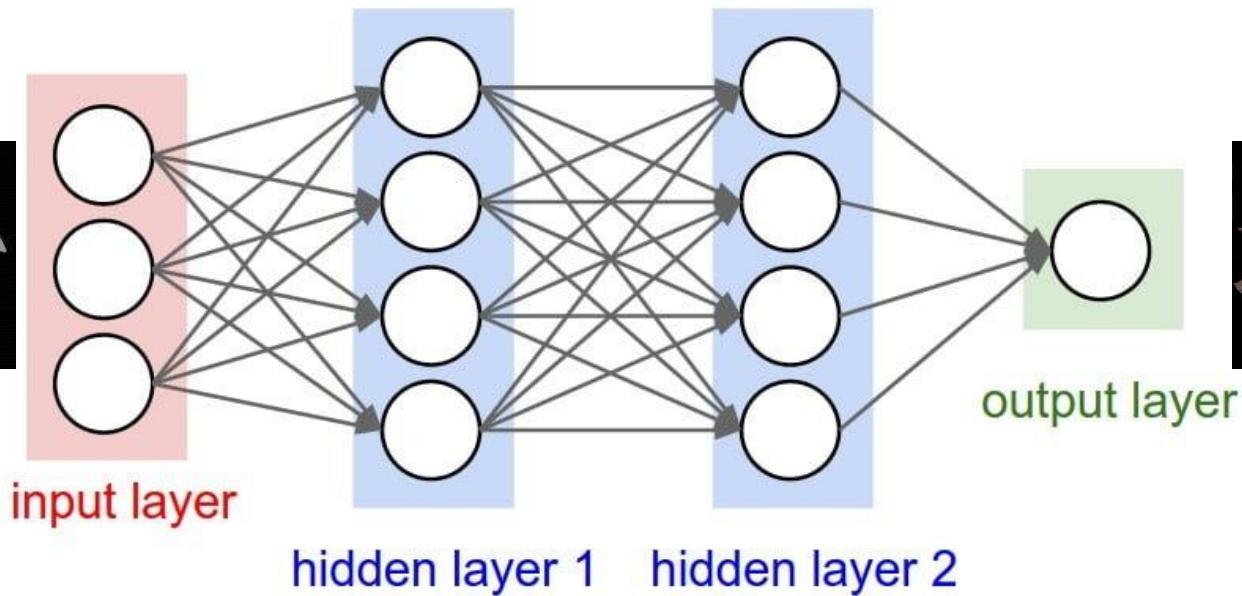
# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



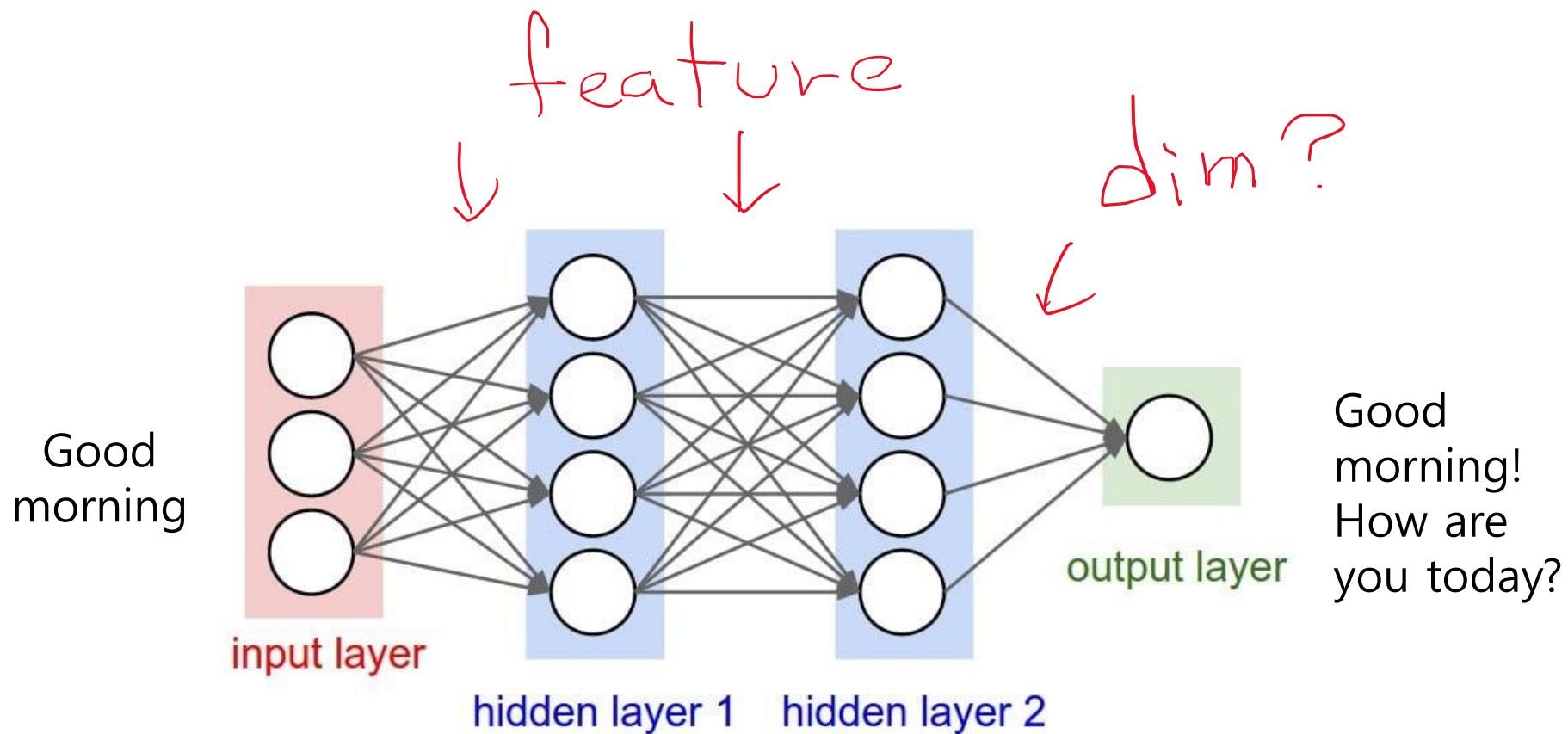
# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



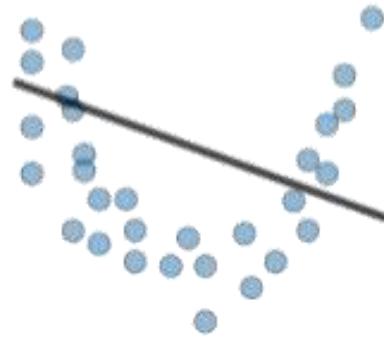
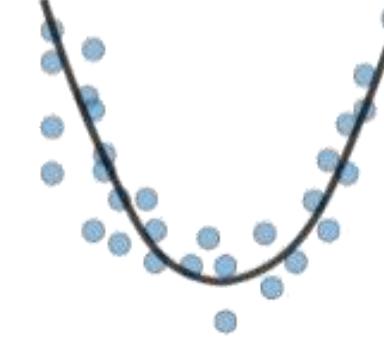
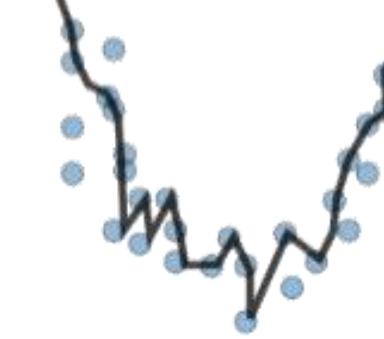
# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)



# 딥러닝의 구조적 이해

신경망 기본 구조 (Perceptron, MLP)

	Underfitting	Just right	Overfitting
Symptoms	<ul style="list-style-type: none"><li>• High training error</li><li>• Training error close to test error</li><li>• High bias</li></ul>	<ul style="list-style-type: none"><li>• Training error slightly lower than test error</li></ul>	<ul style="list-style-type: none"><li>• Very low training error</li><li>• Training error much lower than test error</li><li>• High variance</li></ul>
Regression illustration			
Deep learning illustration			
Possible remedies	<ul style="list-style-type: none"><li>• Complexify model</li><li>• Add more features</li><li>• Train longer</li></ul>		<ul style="list-style-type: none"><li>• Perform regularization</li><li>• Get more data</li></ul>

Abhishek  
Shrivastava, 2020,  
Underfitting Vs Just  
right Vs Overfitting  
in Machine  
learning,  
Kaggle([www.kaggle.com/getting-started/166897](http://www.kaggle.com/getting-started/166897))

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Epoch

손실 함수 값을 최소화하기 위해 모델의 가중치와 편향을 계속해서 업데이트하는 과정. 이 업데이트는 일반적으로 데이터를 작은 묶음인 배치(batch) 단위로 모델에 주입하면서 이루어짐

하나의 에폭은 전체 훈련 데이터셋이 모델의 입력으로 순전파를 통해 예측값을 계산하고, 역전파를 통해 가중치와 편향을 업데이트

## Batch gradient descent (BGD)

경사 하강법(Gradient Descent)의 한 종류로, 모델의 가중치를 업데이트할 때 전체 훈련 데이터셋을 한 번에 사용하여 기울기(gradients)를 계산하는 최적화 알고리즘

### Batch size



1 Epoch

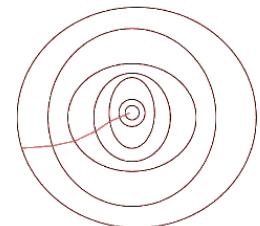
Dataset number = Sample = 10

for index in range(epoch):

$\hat{y} = \text{Tensor}(\text{All dataset}) * \text{NN in GPU}$

Loss = Sum  $(\hat{y} - y) / N$

Back propagation (Loss)



### 장점

1. 안정적인 수렴
2. 노이즈 감소 & 쉬운 수렴 기준

### 단점

1. 느린 계산 속도 & 높은 메모리 사용량
2. 지역 최솟값 문제 & 실용성 부족

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Mini-batch gradient descent (MB-GD, batch size or subdivision)

미니배치 경사 하강법은 모델 학습 시 데이터를 몇 개의 작은 묶음(미니배치)으로 나누어 순차적으로 학습. 각 업데이트 단계에서 하나의 미니배치에 대한 손실 함수의 평균 기울기를 계산하고, 이 기울기를 사용하여 모델의 파라미터를 업데이트. 모든 미니배치를 다 처리하면 1 에폭 완료



Mini-batch size = 2

Dataset = 10, Batch size = 2  
(GD), Steps of 1 Epoch = 5

```
for index in range(epoch):
    for mini_batch in (Batch / subdivision):
         $\hat{y}$ =Tensor(mini_batch.dataset) * NN in GPU
        Loss = Sum ( $\hat{y}$  - y) / N
        Back propagation (Loss) to update NN weights using Gradient Descent (GD)
```

### 장점

1. 효율적인 계산 및 빠른 수렴: 전체 데이터를 사용하는 BGD보다 빠르고, 단일 샘플을 사용하는 SGD보다 안정적임. GPU 병렬 처리에 유리
2. 안정적인 업데이트 및 일반화 성능 향상: 배치 단위 평균 기울기 사용으로 업데이트가 안정적

### 단점

1. 배치 크기 선택: 최적의 배치 크기를 선택하는 것이 중요. 이는 하이퍼파라미터 튜닝의 대상이 됨. 너무 작거나 크면 성능 저하가 발생할 수 있음
2. 지역 최솟값 문제 및 완전한 전역 최솟값 보장 없음: 지역 최솟값에 갇힐 가능성이 있음

# 딥러닝의 구조적 이해

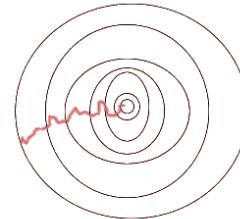
Optimizer, Learning Rate, Batch Size 의미

## Stochastic gradient descent (SGD) 확률적 경사하강

BGD나 MB-GD와 달리, SGD는 매 업데이트마다 무작위 선택된 단 하나 샘플로부터 계산 학습함



Dataset = 10, Sample size = 1



For each epoch:

Shuffle the training dataset

For each single sample in the dataset:

$\hat{y} = \text{Neural\_Network}(\text{sample\_features})$

Loss =  $(\hat{y} - y)^2$

Backpropagate(Loss)

Update\_NN\_Weights()

### 장점

- 매우 빠른 계산 & 높은 효율성
- 지역 최솟값 탈출 용이: 업데이트의 불규칙성(높은 분산)이 오히려 비볼록 손실 함수에서 지역 최솟값에 갇히는 것을 방지, 전역 최솟값 찾을 가능성을 높임
- 메모리 효율성: 한 번에 하나의 샘플만 메모리에 로드하므로, 메모리 사용량이 매우 적음

### 단점

- 불안정한 업데이트: 각 업데이트가 단일 샘플의 노이즈에 크게 영향을 받아, 손실 함수의 감소 경로가 매우 불규칙하고 진동이 심할 수 있음
- 완전한 수렴의 어려움: 손실이 최솟값 근처에서 진동하는 경향이 있어, 완벽하게 수렴하기 어려울 수 있음
- 병렬 처리의 비효율성: 각 업데이트가 독립적인 단일 샘플에 기반하므로, GPU와 같은 병렬 처리 장치의 이점을 충분히 활용하기 어려움

# 딥러닝의 구조적 이해

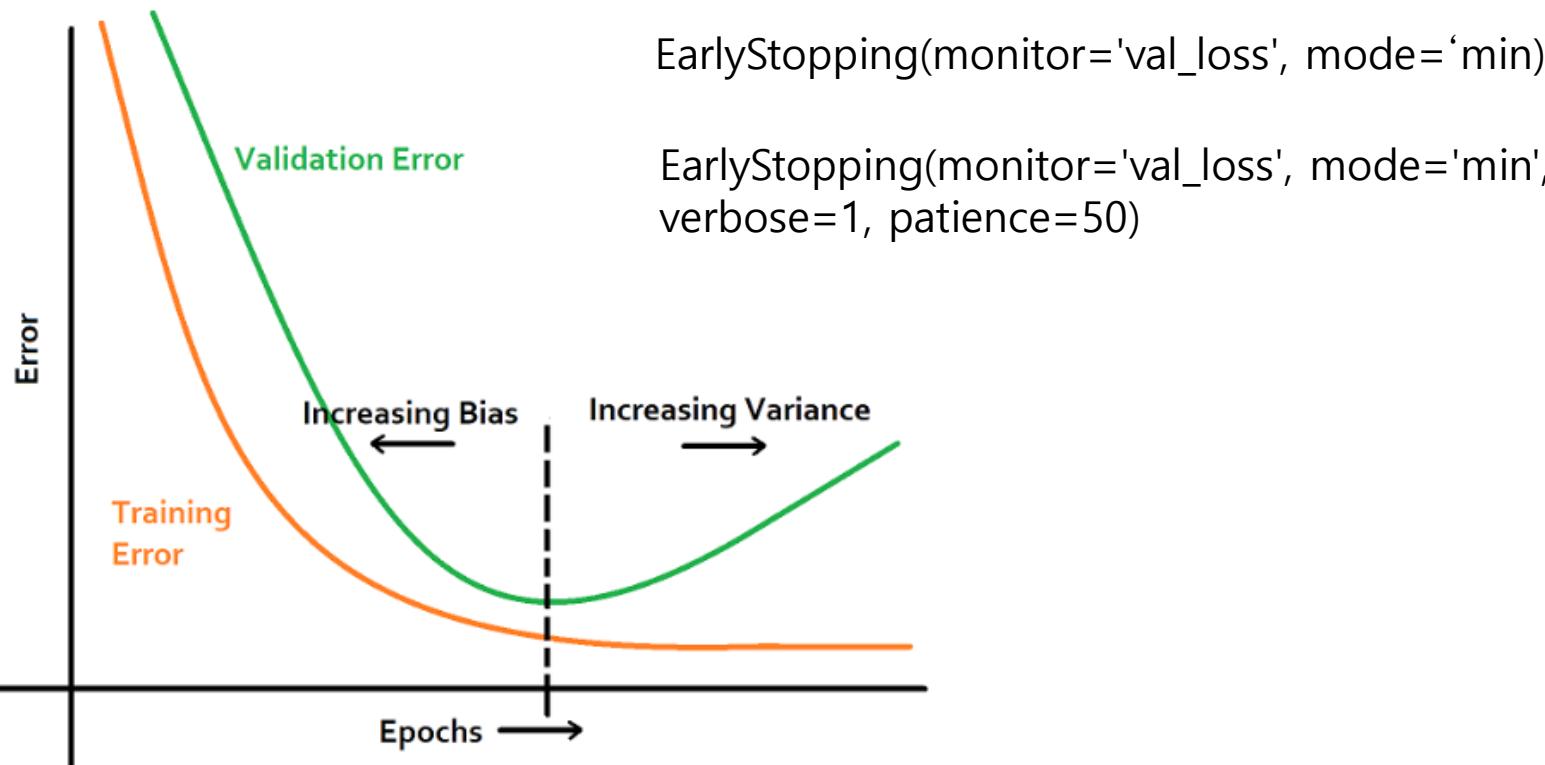
Optimizer, Learning Rate, Batch Size 의미

## 정규화 기법(normalization)

모델이 훈련 데이터에 과적합되는 것을 방지하고, 새로운 데이터에 대한 일반화 성능을 향상시키기 위한 다양한 전략

## Early stopping

조기 종료는 과적합을 방지하기 위해 반복적인 모델 훈련 중에 사용되는 정규화 기법  
훈련 세트에서의 성능이 계속 향상되더라도, 별도의 검증 세트(validation set)에서의 성능이 나빠지기 시작하면 훈련을 멈추는 방식



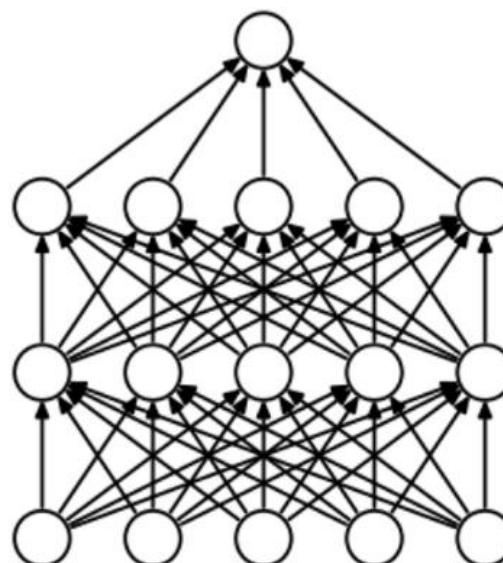
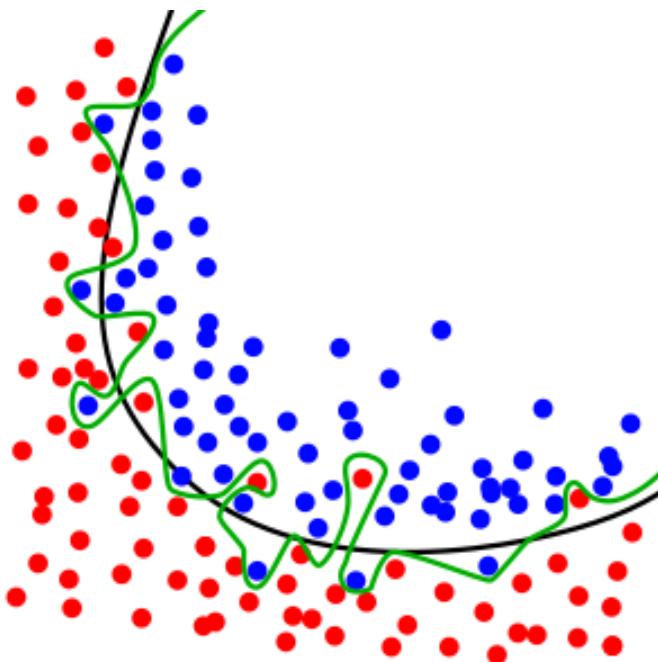
# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

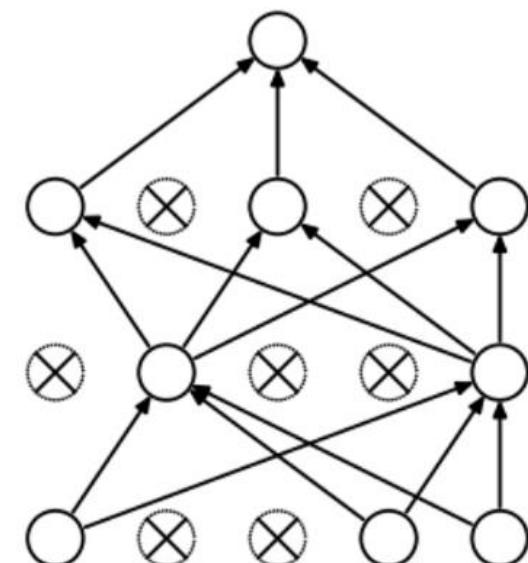
## Dropout

신경망의 과적합을 방지하기 위한 정규화 기법

훈련 과정 동안 신경망의 각 층에 있는 뉴런 중 일부를 무작위로 선택하여 일시적으로 비활성화



(a) Standard Neural Net



(b) After applying dropout.

```
tf.model.add(tf.keras.layers.Dropout(drop_rate))
```

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## L1 Norm (Lasso)

가중치의 절대값의 합을 손실 함수에 추가하여 가중치를 규제. 특정 뉴런의 특성 가중치들이 너무 커지는 것을 방지. 가중치의 절댓값 합에 비례하는 벌칙을 추가, 일부 가중치를 0으로 만들어, 특성 선택(Feature Selection) 효과 발생

$$Cost = \frac{1}{n} \sum_{i=1}^n \left\{ L(y_i, \hat{y}_i) + \frac{\lambda}{2} |w| \right\}$$

정규화 계수  
(regularization strength)

```
Dense(64, activation='relu',  
      kernel_regularizer=regularizers.l1(0.01))
```

## L2 Norm (Ridge)

가중치의 제곱 합에 비례하는 벌칙을 추가. 가중치들이 골고루 작아지도록 유도하여 모델의 복잡도를 줄이고 과적합을 방지

$$Cost = \frac{1}{n} \sum_{i=1}^n \left\{ L(y_i, \hat{y}_i) + \frac{\lambda}{2} |w|^2 \right\}$$

```
Dense(64, activation='relu',  
      kernel_regularizer=regularizers.l2(0.01))
```

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Optimization algorithm

머신러닝 모델의 성능을 향상시키기 위해 모델의 파라미터(가중치와 편향)를 조정하는 방법

### Gradient Descent with Momentum

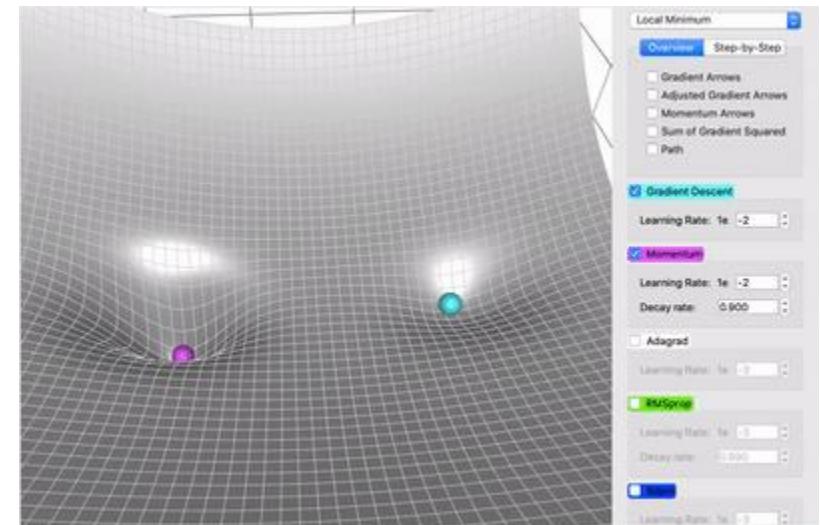
일반적인 경사 하강법은 현재 위치에서 손실 함수가 가장 가파르게 감소하는 방향(음의 기울기)으로 이동. 이 방법은 지그재그 현상 발생 가능. 손실 함수 표면이 길고 좁은 계곡 형태일 때, 최솟값을 향해 직선으로 내려가지 못하고 좌우로 심하게 진동 가능

이전 파라메터 업데이트값의 관성(inertia)을 사용

$$V(t) = m * V(t - 1) - \alpha \frac{\partial}{\partial w} Cost(w)$$
$$W(t + 1) = W(t) + V(t)$$

m (모멘텀 계수): 0과 1 사이의 값

$V(t-1)$ : 이전 시간 단계  $t-1$ 에서 속도(모멘텀 벡터)  
if  $t=1$  then  $V=0$



# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## ADAM(Adaptive moment estimation)

경사 하강법 기반 최적 알고리즘. 모델 훈련 시 가중치와 편향을 효율적으로 업데이트

**Momentum (모멘텀):** 이전 기울기들의 지수 가중 이동

평균을 계산하여, 가중치 업데이트 방향을 결정

최적화 과정에서 불필요한 진동을 줄임. 지역 최솟값을 넘어설 수 있도록 관성을 부여하는 효과 발생

**RMSProp (Root Mean Square Propagation):** 파라메터

기울기 제곱의 지수의 가중 이동 평균을 계산하여, 이를 학습률 조정하는 데 사용. 기울기 큰 파라미터는 학습률을 줄이고, 작으면 학습률을 늘려 안정적 학습 유도

**편향 보정 (Bias Correction):** 1차 모멘트  $m$ 과 2차 모멘트  $v$  계산시,  $t=0$ 일 때  $m=0$ ,  $v=0$  임. 이 초기값 0이 강한 편향을 유발해, 불안정해지는 가중치 갱신을 안정화 시킴

$$M(t) = \beta_1 M(t-1) + (1 - \beta_1) \frac{\partial}{\partial w(t)} Cost(w(t))$$

$$V(t) = \beta_2 V(t-1) + (1 - \beta_2) \left( \frac{\partial}{\partial w(i)} Cost(w(i)) \right)^2$$

$$\hat{M}(t) = \frac{M(t)}{1 - \beta_1^t} \quad \hat{V}(t) = \frac{V(t)}{1 - \beta_2^t}$$

$$W(t+1) = W(t) - \alpha * \frac{\hat{M}(t)}{\sqrt{\hat{V}(t) + \epsilon}}$$

$$m_1 = \beta_1 m_0 + (1 - \beta_1) g_1$$

$$m_1 = 0.9 \cdot 0 + (1 - 0.9) \cdot 5$$

$$m_1 = 0 + 0.1 \cdot 5 = \mathbf{0.5}$$

$$v_1 = \beta_2 v_0 + (1 - \beta_2) g_1^2$$

$$v_1 = 0.999 \cdot 0 + (1 - 0.999) \cdot 5^2$$

$$v_1 = 0 + 0.001 \cdot 25 = \mathbf{0.025}$$

$$\hat{m}_1 = \frac{m_1}{1 - \beta_1^t} = \frac{0.5}{1 - 0.9^1} = \frac{0.5}{1 - 0.9} = \frac{0.5}{0.1} = \mathbf{5.0}$$

$$\hat{v}_1 = \frac{v_1}{1 - \beta_2^t} = \frac{0.025}{1 - 0.999^1} = \frac{0.025}{1 - 0.999} = \frac{0.025}{0.001} = \mathbf{25.0}$$

$$W_{1+1} = 10 - 0.01 \cdot \frac{5.0}{\sqrt{25.0 + 10^{-8}}}$$

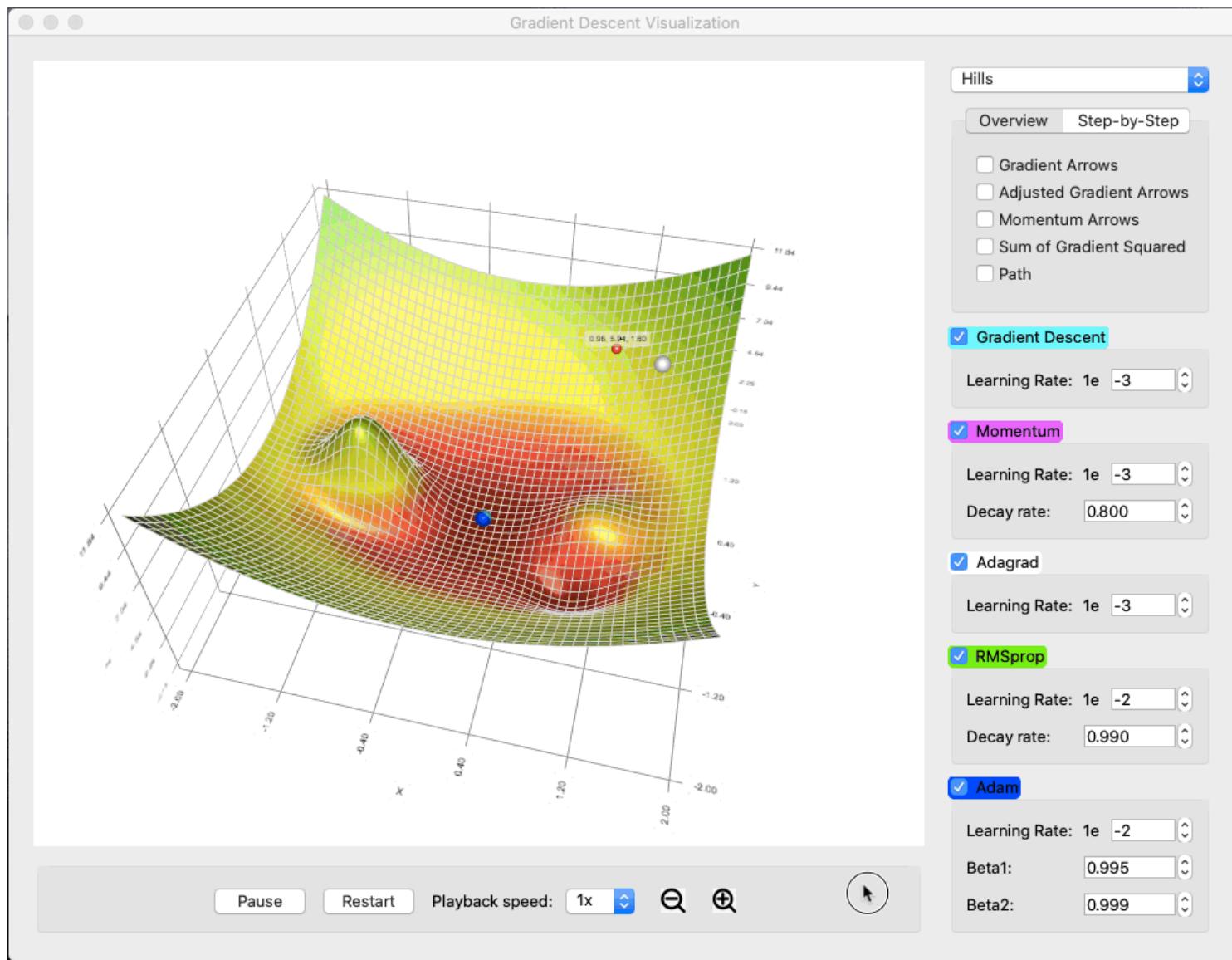
$$W_2 = 10 - 0.01 \cdot \frac{5.0}{5.0 + 10^{-8}}$$

$$W_2 \approx 10 - 0.01 \cdot 1.0 = \mathbf{9.99}$$

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Gradient Descent Method Simulation



# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

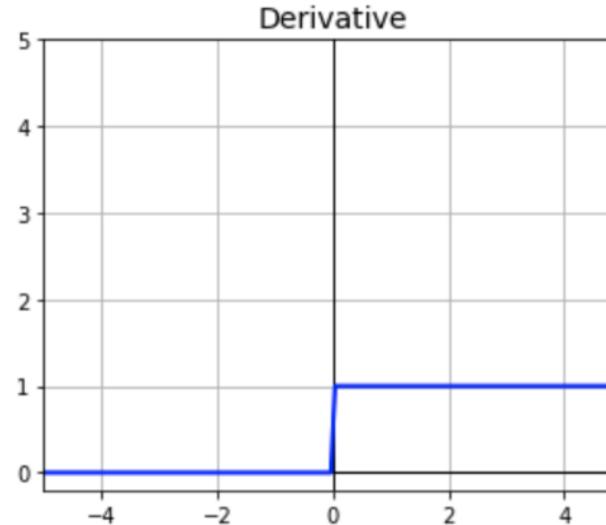
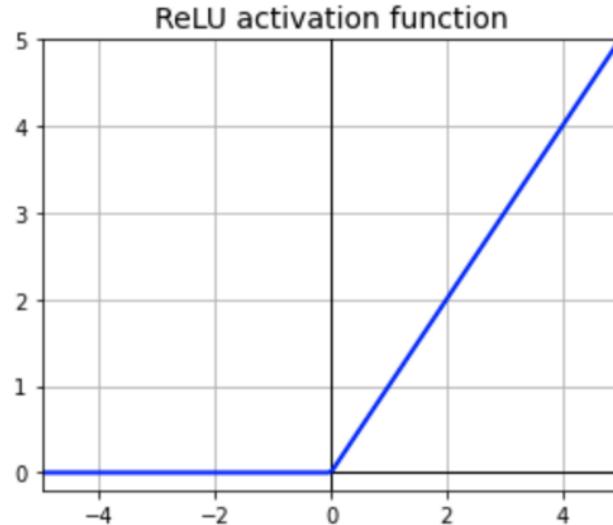
## ReLU (Rectified Linear Unit)

계산 쉬워 속도 빠름. 양수 영역 기울기는 항상 1, 역전파 과정에서 기울기 0이 되어 학습이 멈추는 기울기 소실 문제 방지. 음수입력은 0을 출력, 뉴런 비활성화하여 Dropout 효과 발생 단, Dying ReLU 문제 발생해 이 뉴런과 연결된 가중치 업데이트 안되는 문제도 발생

$$f(x) = \max(0, x)$$

$$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$

```
def relu(x):  
    if x < 0:  
        return 0  
    else:  
        return x
```



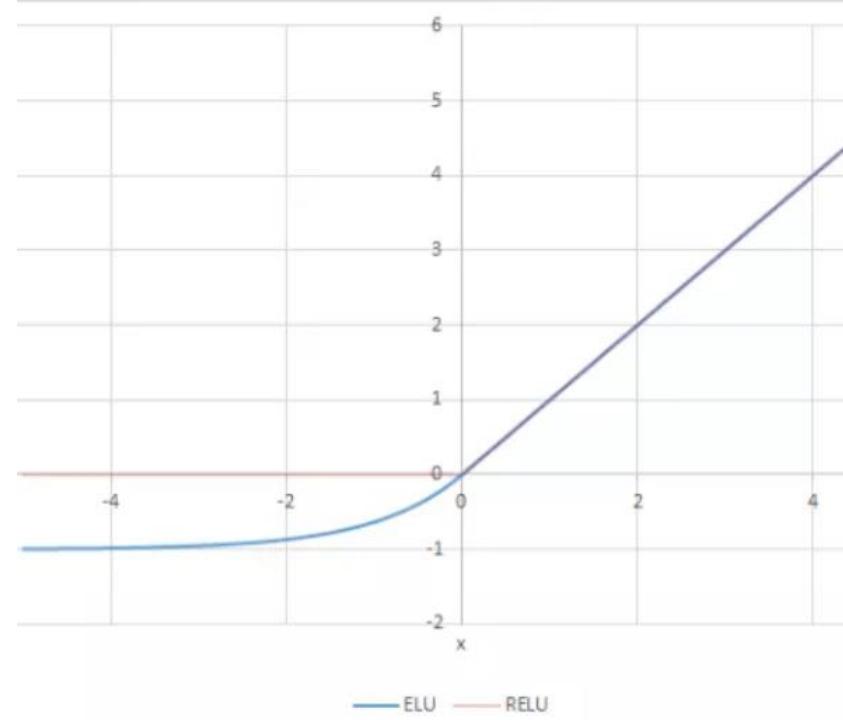
# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Exponential Linear Unit (ELU)

음수 입력에 대해서도 작은 음수 값을 출력, Dying ReLU처럼 뉴런 비활성화 문제 방지

$$R(z) = \begin{cases} z & z > 0 \\ \alpha \cdot (e^z - 1) & z \leq 0 \end{cases}$$



```
def elu(z,alpha):  
    return z if z >= 0 else alpha*(e^z - 1)
```

# 딥러닝의 구조적 이해

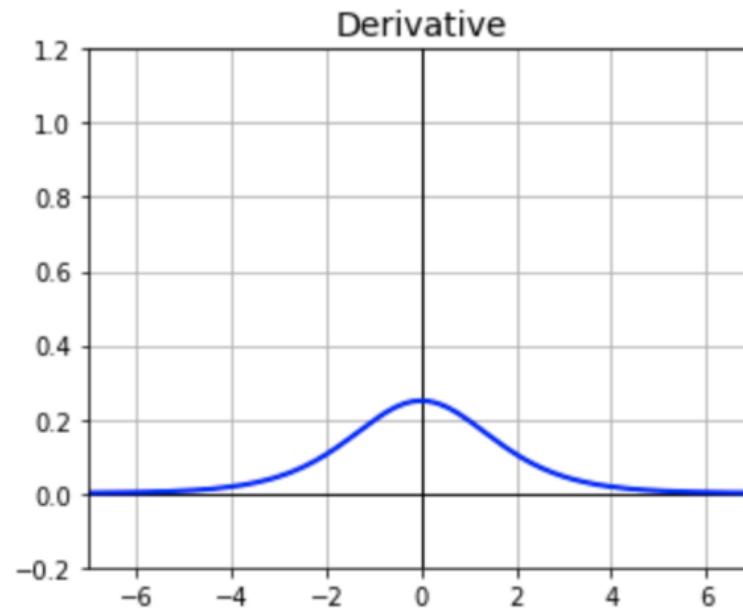
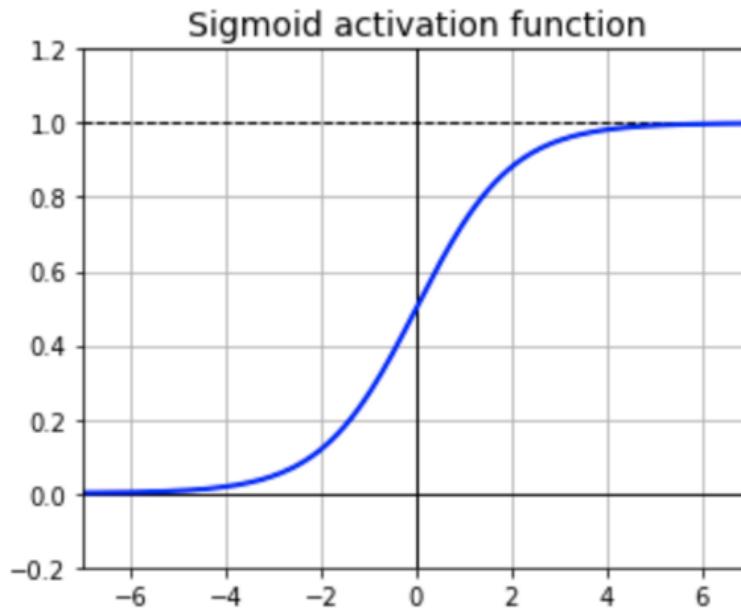
Optimizer, Learning Rate, Batch Size 의미

## Sigmoid

출력을 0과 1사이로 정규화, 미분 가능. 계산 비용 높음. 이진 분류 문제 등에 유리. 단, 입력값이 매우 크거나 작을 때 미분한 기울기 값이 0에 가까워져 기울기 소실 문제 발생. 출력 평균이 0이 아니므로 편향된 가중치 갱신, 이로 인한 늦은 학습 수렴 속도

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Vanishing gradient  
Computationally expensive  
The output is not zero centered

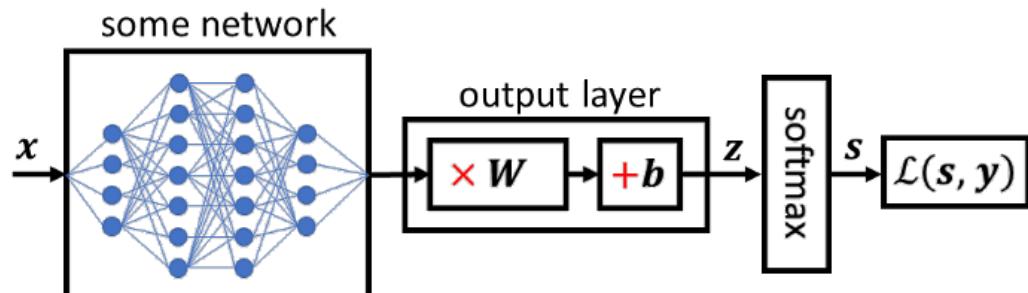
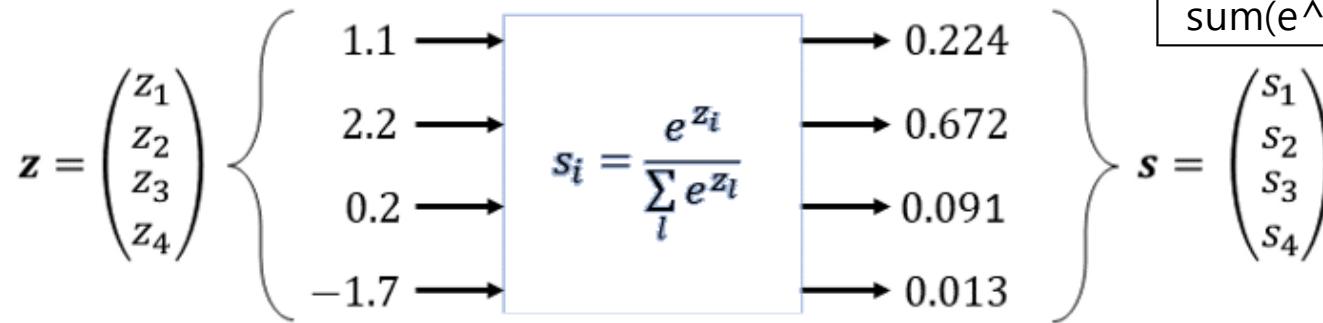


# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

Softmax

$$s_i = \frac{e^{z_i}}{\sum_{l=1}^n e^{z_l}}$$



$$-1.7 < 0.2 < 1.1 < 2.2 \rightarrow 0.013 < 0.091 < 0.224 < 0.672$$

# 딥러닝의 구조적 이해

## Loss 함수

### 손실 함수

MSE(Mean Square Error)

$$MSE = \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}$$

### 장점

- 미분 가능하고 최적화 용이
- 큰 오차에 대한 패널티: 오차를 제곱하여 큰 오차에 더 큰 패널티를 부여

### 단점

이상치에 민감함: 큰 오차에 대한 패널티가 커서 이상치에 취약

MAE(Mean Absolute Error)

$$MAE = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n}$$

- 이상치에 강건: 오차에 절댓값을 취하여 이상치에 덜 민감
- 직관적인 해석: 손실 값의 단위가 실제 예측 단위와 동일, 해석이 직관적

1. 미분 불가능 지점 존재: 오차 0인 지점에서 미분 불가능. 최적화에 어려움

2. 느린 수렴: 기울기 크기가 오차 크기에 관계없이 일정, 최솟값 근처에서 수렴이 느릴 수 있음

RMSE(Root Mean Square Error)

$$RMSE = \sqrt{\sum_{i=1}^n \frac{(\hat{y}_i - y_i)^2}{n}}$$

- 단위 해석 용이함: MSE와 달리 제곱근을 취하여 단위가 실제 예측 단위와 동일하여 해석이 용이
- 큰 오차에 대한 패널티 유지함: MSE의 큰 오차 패널티 부여 특성을 유지

이상치에 민감함: MSE에서 파생된 값이므로 여전히 이상치에 민감함

# 딥러닝의 구조적 이해

## Loss 함수

### 손실 함수

BCE(Binary Cross Entropy Loss)

$$L(y, \hat{y}) = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y}))$$

$$L = -(1 \cdot \log(0.9) + (1 - 1) \cdot \log(1 - 0.9))$$

$$L = -(1 \cdot \log(0.9) + 0 \cdot \log(0.1))$$

$$L = -\log(0.9)$$

$$L \approx -(-0.1053)$$

$$L \approx \underline{\underline{0.1053}}$$

### 장점

- 확률 분포 차이 측정 적합: 예측 확률이 실제 정답 분포와 얼마나 잘 일치하는지 측정하는데 수학적으로 적합
- 수렴에 유리: 모델이 잘못 예측할수록 손실이 가파르게 증가하여 초기 단계에서 모델이 빠르게 학습하여 수렴
- 미분 가능함: 모든 지점에서 미분 가능하며, Sigmoid (0~1) 함께 사용될 때 기울기 소실 문제가 완화, 학습이 효율적

### 단점

- 클래스 불균형에 취약: 긍정/부정 클래스 비율이 불균형할 경우, 모델이 다수 클래스 예측에 편향
- 직관적이지 않은 해석: MSE나 MAE처럼 '오차의 평균'과 같은 직관적인 해석 어려움

CCE(Categorical Cross-Entropy)

$$L(y, \hat{y}) = -\sum_{k=1}^C y_k \log(\hat{y}_k)$$

$C$ : 총 클래스 수

$y_k$ : 실제 정답을 원-핫 인코딩한 값 (정답 클래스면 1, 아니면 0)

$\hat{y}_k$ : 모델이 예측한  $k$ 번째 클래스의 확률

[1, 0, 0] (고양이)

[0.9, 0.05, 0.05] (고양이 90%, 강아지 5%, 새 5%)

$$L = -(y_0 \log(\hat{y}_0) + y_1 \log(\hat{y}_1) + y_2 \log(\hat{y}_2))$$

$$L = -(1 \cdot \log(0.9) + 0 \cdot \log(0.05) + 0 \cdot \log(0.05))$$

$$L = -\log(0.9)$$

$$L \approx -(-0.1053)$$

$$L \approx \underline{\underline{0.1053}}$$

- 여러 클래스 분류 가능: 세 개 이상의 상호 배타적인 클래스 효과적 분류
- 확률적 해석 제공: Softmax 출력과 함께 각 클래스에 속할 확률 분포 제공
- 미분 가능 및 최적화 용이: 손실 함수가 미분 가능, 경사 하강법 기반 최적화에 적합

- 클래스 불균형에 취약: 특정 클래스 데이터가 압도적으로 많을 경우 다수 클래스에 편향되어 소수 클래스 성능이 저하
- 원-핫 인코딩 필수: 클래스 수만큼 softmax 처리 출력 필요. 정답 레이블이 원-핫 인코딩 형태로 준비되어야 함

# 딥러닝의 구조적 이해

## Loss 함수

Loss 함수	용도
MSE (Mean Squared Error)	회귀(Regression) 문제: 연속적인 숫자 값 예측 시 사용 (예: 주택 가격 예측, 온도 예측). 오차의 크기가 중요하고 큰 오차에 더 큰 패널티를 줄 때 적합
MAE (Mean Absolute Error)	회귀(Regression) 문제: 연속적인 숫자 값 예측 시 사용 (예: 주택 가격 예측, 온도 예측). 이상치(Outlier)에 덜 민감하게 반응하고 싶을 때 적합
RMSE (Root Mean Squared Error)	회귀(Regression) 문제: MSE와 유사하게 연속적인 숫자 값 예측 시 사용. MSE의 단위를 실제 예측 단위와 동일하게 맞춰 해석의 용이성을 높이고 싶을 때 사용
BCE (Binary Cross-Entropy)	이진 분류(Binary Classification) 문제: 두 가지 상호 배타적인 클래스 중 하나로 분류할 때 사용 (예: 스팸/일반 메일 분류, 질병 유무 예측). 출력층에 시그모이드 (Sigmoid) 활성화 함수를 사용
CCE (Categorical Cross-Entropy)	다중 클래스 분류(Multi-class Classification) 문제: 세 개 이상의 상호 배타적인 클래스 중 하나로 분류할 때 사용 (예: 손글씨 숫자 인식, 이미지 속 객체 분류). 출력 층에 소프트맥스(Softmax) 활성화 함수를 사용

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Learning Rate Decay

훈련 진행됨에 따라 학습률 값을 점진적으로 줄여나가는 전략. 모델 훈련의 안정성과 성능을 향상시키기 위해 사용

Time-based decay

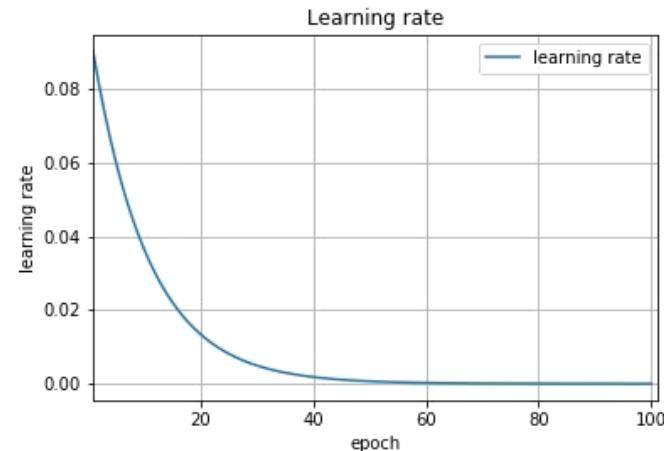
$$\alpha_t = \frac{\alpha_0}{1+k \cdot t}$$

learning\_rate = 0.1

decay\_rate = learning\_rate / epochs

momentum = 0.8

optimization = SGD(lr=learning\_rate, momentum=momentum, decay=decay\_rate)



Step decay

$$\alpha(t) = \alpha_0 \cdot \gamma^{\lfloor \frac{t}{s} \rfloor} \quad \lfloor \cdot \rfloor = \text{floor}(), s = \text{drop num}, t = \text{epoch}$$

def step\_decay(epoch):

    initial\_lrate = 0.1

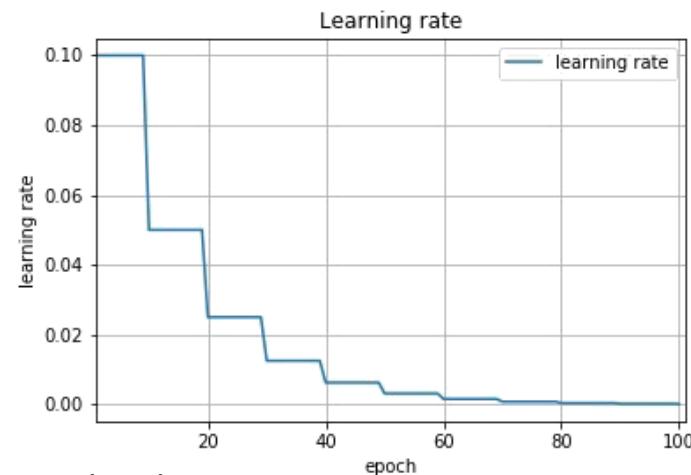
    drop = 0.5

    epochs\_drop = 10.0

    lrate = initial\_lrate \* math.pow(drop, math.floor((1+epoch)/epochs\_drop))

    return lrate

lrate = LearningRateScheduler(step\_decay)



# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Batch normalization

배치 정규화(batchnorm)는 신경망 성능 정확도를 향상.  
배치 정규화는 배치별로 발생

네트워크에 대한 입력을 정규화하는 대신 네트워크 내각 숨겨진 계층에 대한 입력을 정규화함

훈련 중에 들어오는 미니배치 데이터의 평균을 0, 분산을 1로 정규화 후 학습 가능한 스케일(gamma), 쉬프트(beta) 변수로 조정함

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_{1\dots m}\}$ ;  
Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = BN_{\gamma, \beta}(x_i)\}$

$$\begin{aligned}\mu_{\mathcal{B}} &\leftarrow \frac{1}{m} \sum_{i=1}^m x_i && // \text{mini-batch mean} \\ \sigma_{\mathcal{B}}^2 &\leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 && // \text{mini-batch variance} \\ \hat{x}_i &\leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} && // \text{normalize} \\ y_i &\leftarrow \gamma \hat{x}_i + \beta \equiv BN_{\gamma, \beta}(x_i) && // \text{scale and shift}\end{aligned}$$

```
from keras.models import Sequential  
from keras.layers import Dense, Dropout  
from keras.layers import BatchNormalization  
...
```

```
y = to_categorical(y)  
n_train = 800  
trainX, testX = X[:n_train, :], X[n_train:, :]  
trainy, testy = y[:n_train], y[n_train:]
```

```
model = Sequential()  
model.add(Dense(25, input_dim=2, activation='relu'))  
model.add(BatchNormalization())  
model.add(Dropout(0.2))  
model.add(Dense(8, activation='relu'))  
model.add(BatchNormalization())  
model.add(Dropout(0.2))  
model.add(Dense(4, activation='softmax'))  
model.compile(loss='categorical_crossentropy',  
optimizer='adam', metrics=['accuracy'])
```

```
model.summary()
```

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

## Data Augmentation

학습 데이터 다양성 확보하여 모델의 일반화 능력 상향 시킴



```
from keras.preprocessing.image import ImageDataGenerator  
from keras.utils import array_to_img, img_to_array, load_img
```

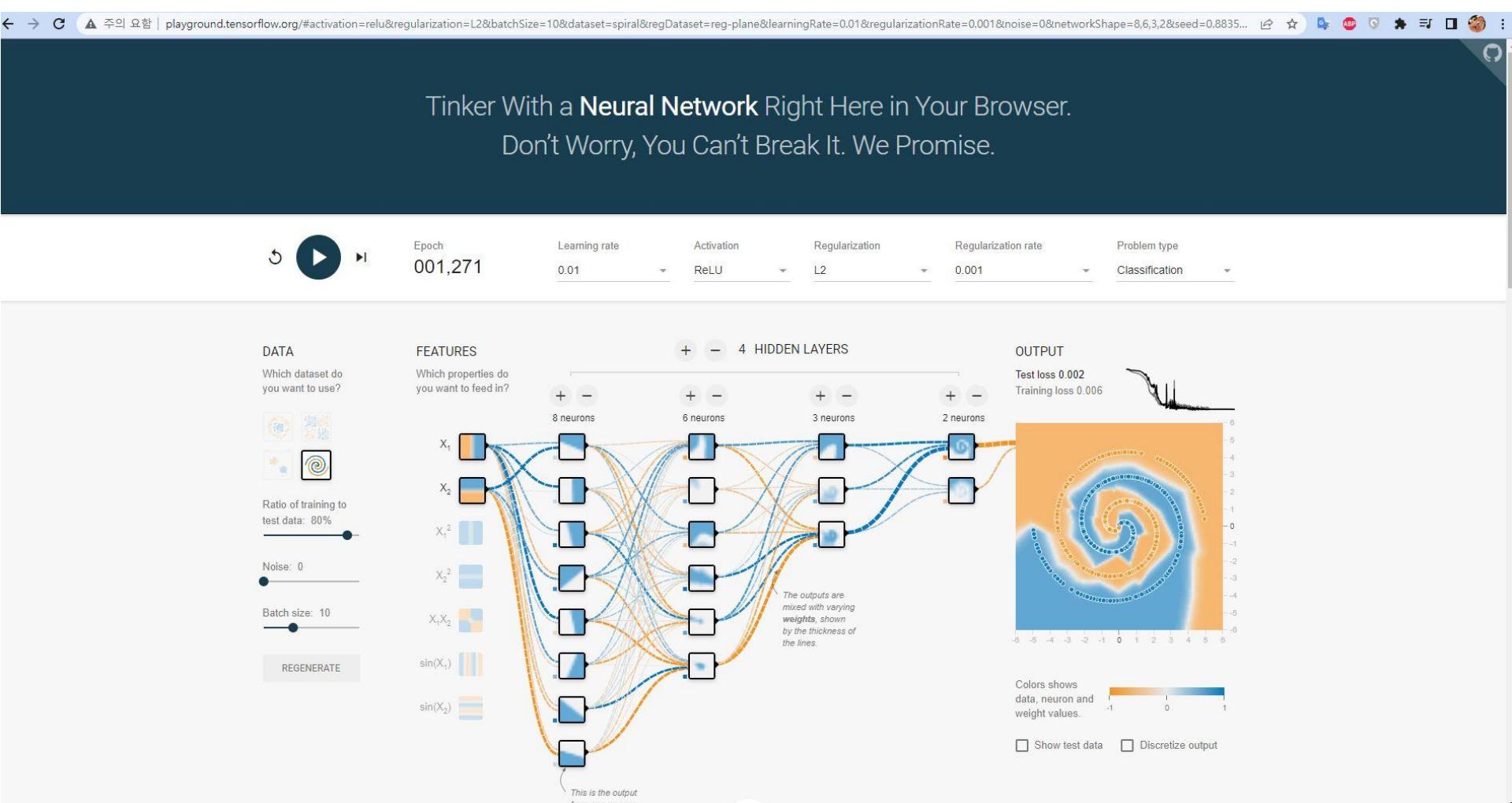
```
datagen = ImageDataGenerator(  
    rotation_range=40, width_shift_range=0.2, height_shift_range=0.2, shear_range=0.2, zoom_range=0.2,  
    horizontal_flip=True, fill_mode='nearest')
```

```
img = load_img('./cat0.jpg')          # this is a PIL image  
x = img_to_array(img)                # this is a Numpy array with shape (3, 150, 150)  
x = x.reshape((1,) + x.shape)        # this is a Numpy array with shape (1, 3, 150, 150)
```

```
i = 0  
for batch in datagen.flow(x, batch_size=1, save_to_dir='preview', save_prefix='cat', save_format='jpeg'):   
    i += 1  
    if i > 20:  
        break # otherwise the generator would loop indefinitely
```

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미



Link

Tinker With a Neural Network

<http://playground.tensorflow.org/#activation=relu&regularization=L2&batchSize=10&dataset=spiral&regDataset=reg-plane&learningRate=0.01&regularizationRate=0.001&noise=0&networkShape=8,6,3,2&seed=0.88358&showTestData=false&discretize=false&percTrainData=80&x=true&y=true&xTimesY=false&xSquared=false&ySquared=false&cosX=false&sinX=false&cosY=false&sinY=false&collectStats=false&problem=classification&initZero=false&hideText=false>

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

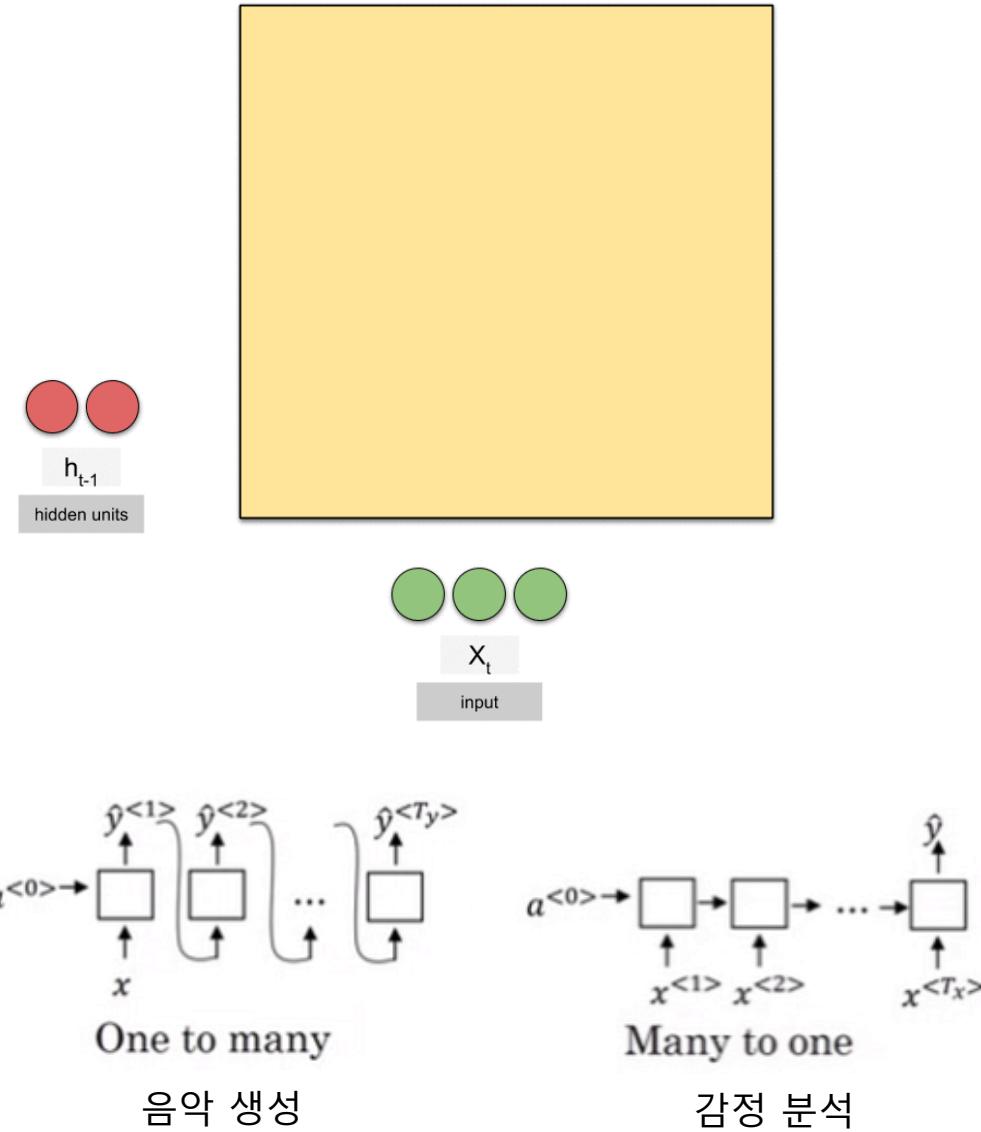
## RNN(Recurrent Neural Network)

순환 신경망 RNN은 Apple의 Siri 및 Google의 음성 검색에서 사용되었음. RNN은 내부 메모리가 있는 피드포워드 신경망의 일반화된 구조

RNN은 모든 데이터 입력에 대해 동일한 기능을 수행하는 반면, 현재 입력의 출력은 과거 한 번의 계산에 따라 달라지기 때문에 본질적으로 반복적임

출력을 생성한 후 복사되어 순환 네트워크로 다시 전송. 최종 출력은 현재 입력과 이전 입력에서 학습한 출력을 고려. RNN에는 메모리 기능 있어, 이전 데이터를 기억. 현재 입력 및 이전 상태를 사용, 현재 상태를 계산

RNN은 그래디언트 소실(Vanishing Gradient) 문제 발생할 수 있음



[Animated RNN, LSTM and GRU. Recurrent neural network cells in GIFs](#)  
| by Raimi Karim | TDS Archive | Medium

[Understanding Recurrent Neural Network \(RNN\) and Long Short Term Memory\(LSTM\)](#) | by Vijay Choubey | Analytics Vidhya | Medium

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

## LSTM (Long Short Term Memory)

RNN에서 새로운 정보가 추가되면 RNN은 기존 정보를 크게 수정하는 문제 발생. LSTM은 입력, 출력 및 망각 게이트를 통해 어떤 데이터가 중요하고 미래에 유용할 수 있는지, 어떤 데이터를 지워야 하는지를 결정 가능

### 입력 게이트 $i_t$

현재 입력  $x_t$ 와 이전 은닉  $h_{t-1}$ 로 부터 새 정보를 셀 상태 저장할 것인지 결정. 새로운 후보 셀  $\tilde{C}_t$  계산해 곱해 출력

### 망각 게이트 $f_t$

이전 셀 상태에서 정보 유지 망각 여부 결정  
출력 sigmoid(0~1) 값은 이전  $C_{t-1}$ 과 곱해짐

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

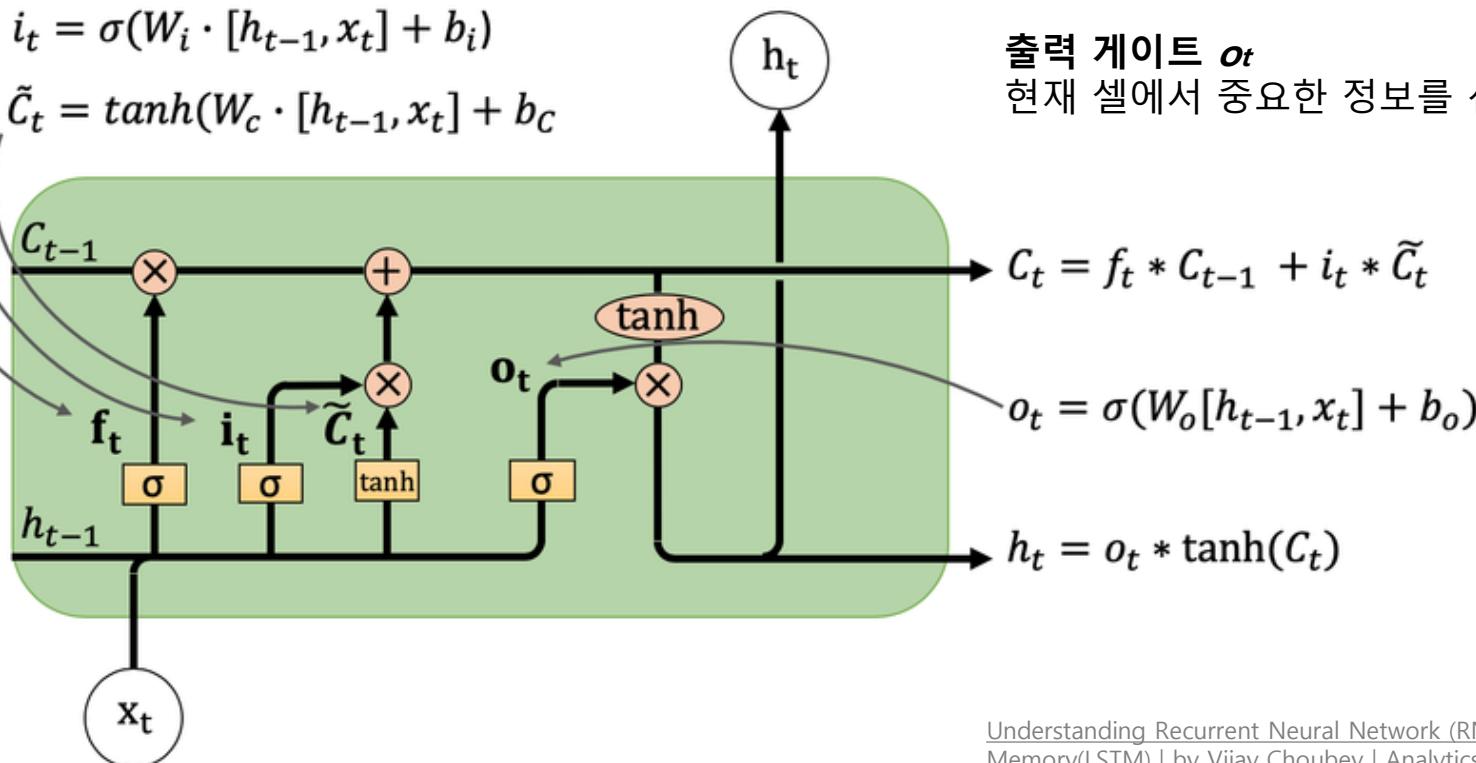
$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

### 셀 상태 업데이트 $C_t$

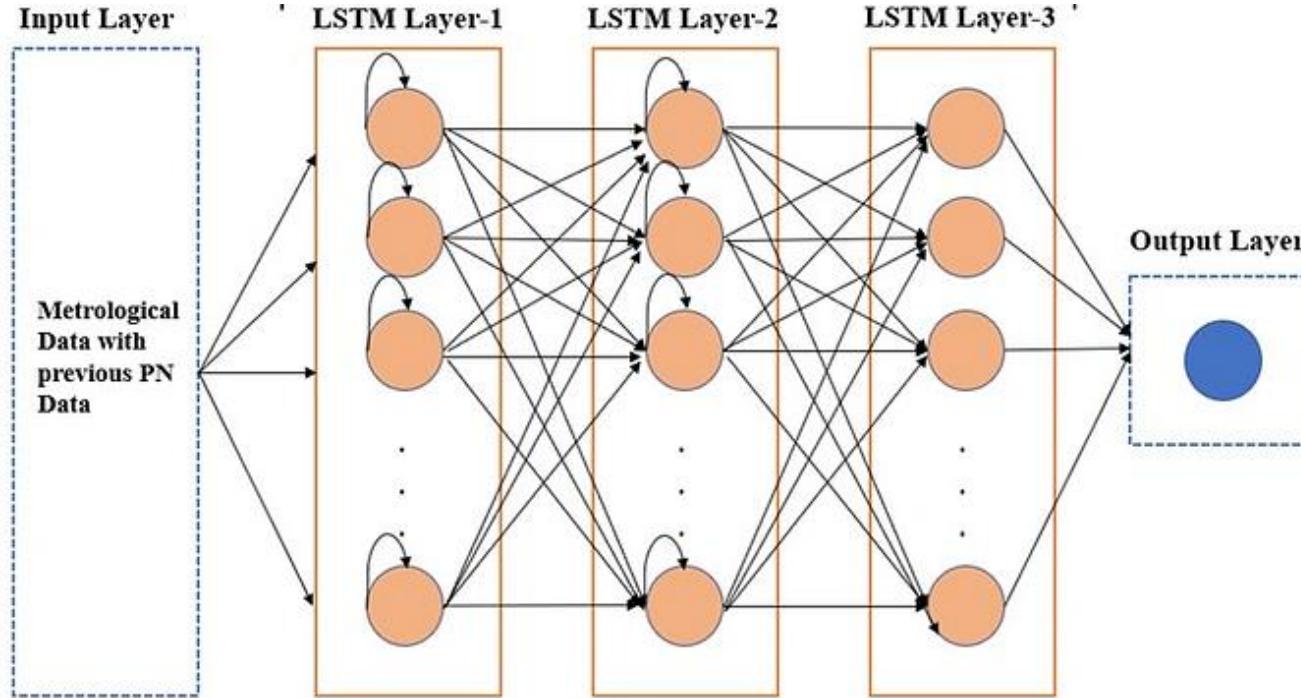
입력, 망각 게이트 정보 더해, 새로운  $C_t$  계산

### 출력 게이트 $o_t$

현재 셀에서 중요한 정보를 선택하여 출력



### LSTM (Long Short Term Memory)



```
from keras.models import Sequential  
from keras.layers import LSTM, Dense, Embedding, SpatialDropout1D
```

```
model = Sequential()  
...  
model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))  
model.add(Dense(1, activation='sigmoid'))
```

[Complete Guide to Learn LSTM Models: Types, Applications, and When to Use Which Model | by Poornam Sai G | Stackademic](#)

## ImageNet Classification with Deep Convolutional Neural Networks



**Alex Krizhevsky**  
University of Toronto  
[kriz@cs.utoronto.ca](mailto:kriz@cs.utoronto.ca)

**Ilya Sutskever**  
University of Toronto  
[ilya@cs.utoronto.ca](mailto:ilya@cs.utoronto.ca)

**Geoffrey E. Hinton**  
University of Toronto  
[hinton@cs.utoronto.ca](mailto:hinton@cs.utoronto.ca)

### Abstract

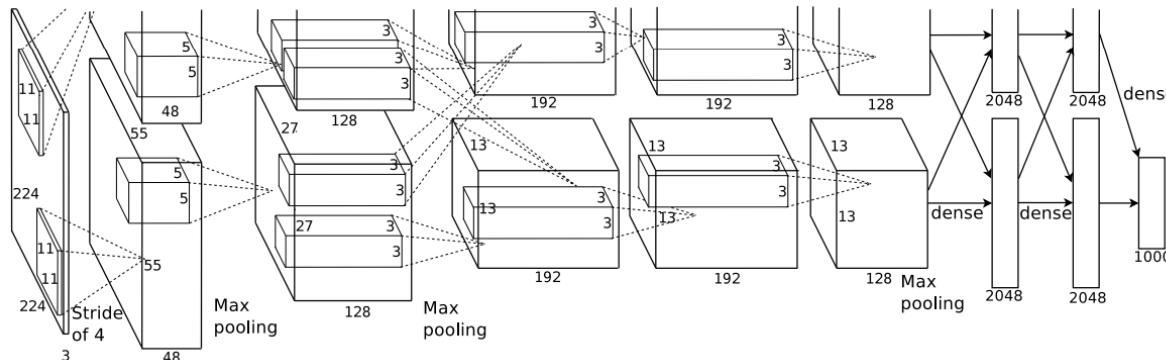


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

## CNN (Convolutional Neural Network)

합성곱 신경망(ConvNets 또는 CNN)은 이미지 인식, 이미지 분류를 수행하는 주요 범주 중 하나

CNN 모델은 각 입력 이미지를 필터(Kernels), 풀링, 완전 연결 계층(FC)이 있는 일련의 컨볼루션 레이어를 통과하고 Softmax 기능을 적용하여 확률 값이 0과 1 사이인 객체를 분류

Filter kernel

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0



1	0	1
0	1	0
1	0	1

5 x 5 – Image Matrix

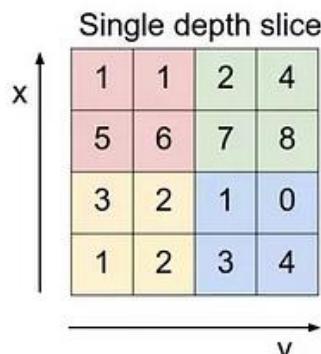
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved Feature

Max pooling



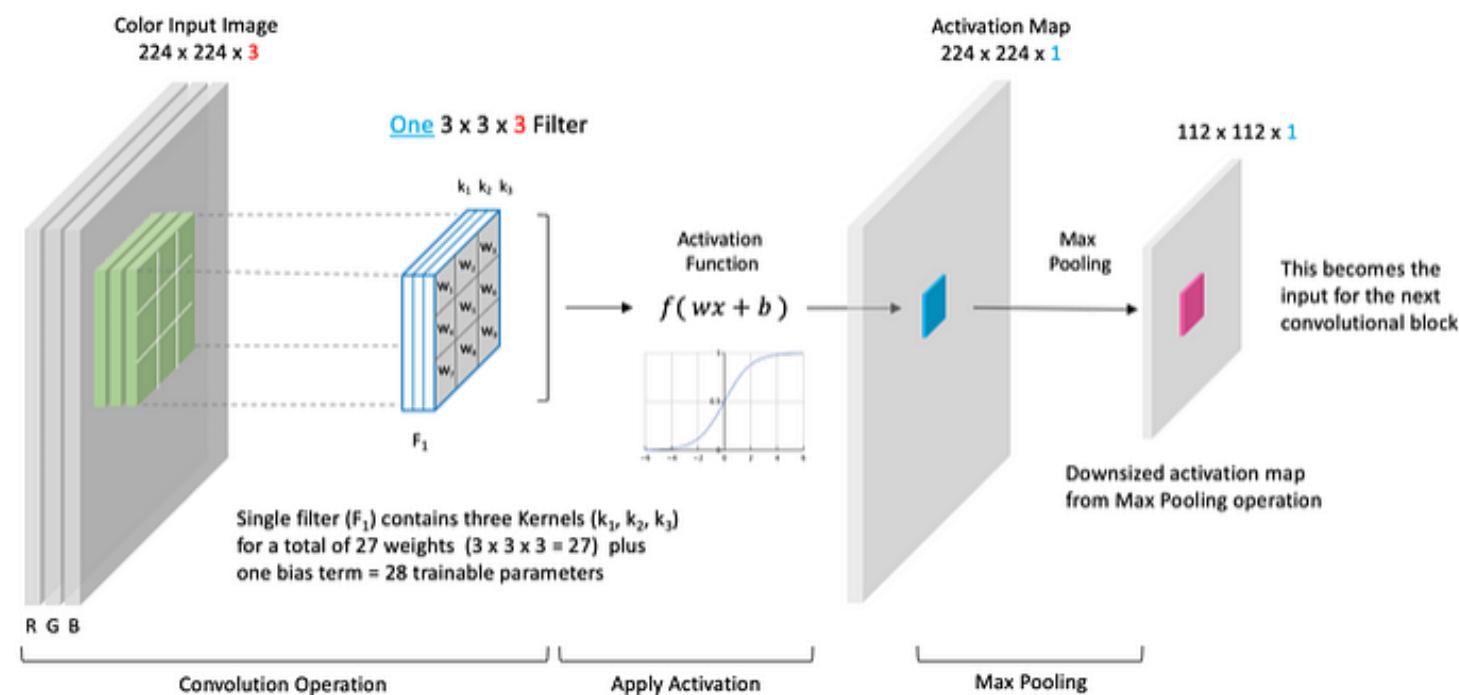
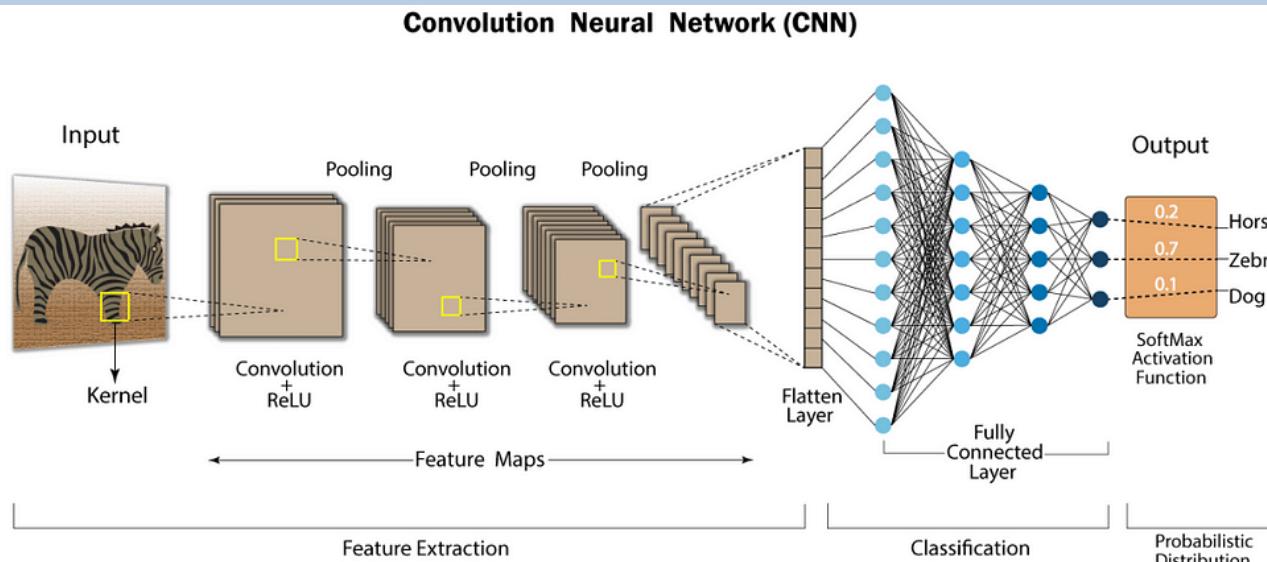
max pool with 2x2 filters  
and stride 2

6	8
3	4

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

## CNN



# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

0	0	0	0	0	0	...
0	156	155	156	158	158	...
0	153	154	157	159	159	...
0	149	151	155	158	159	...
0	146	146	149	153	158	...
0	145	143	143	148	158	...
...	...	...	...	...	...	...

Input Channel #1 (Red)

0	0	0	0	0	0	...
0	167	166	167	169	169	...
0	164	165	168	170	170	...
0	160	162	166	169	170	...
0	156	156	159	163	168	...
0	155	153	153	158	168	...
...	...	...	...	...	...	...

Input Channel #2 (Green)

0	0	0	0	0	0	...
0	163	162	163	165	165	...
0	160	161	164	166	166	...
0	156	158	162	165	166	...
0	155	155	158	162	167	...
0	154	152	152	157	167	...
...	...	...	...	...	...	...

Input Channel #3 (Blue)

-1	-1	1
0	1	-1
0	1	1

Kernel Channel #1

1	0	0
1	-1	-1
1	0	-1

Kernel Channel #2

0	1	1
0	1	0
1	-1	1

Kernel Channel #3

308

+

-498

+

164

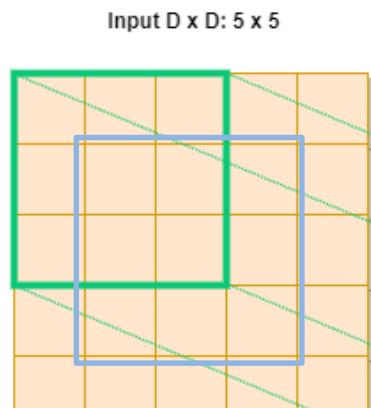
Bias = 1

-25				...
				...
				...
				...
...	...	...	...	...

Output

# 딥러닝의 구조적 이해

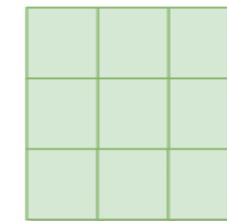
CNN/RNN/LSTM의 개념적 차이



✓ Padding VALID  
Output dimension =  $D - N + 1$   
 $5 - 3 + 1 = 3$

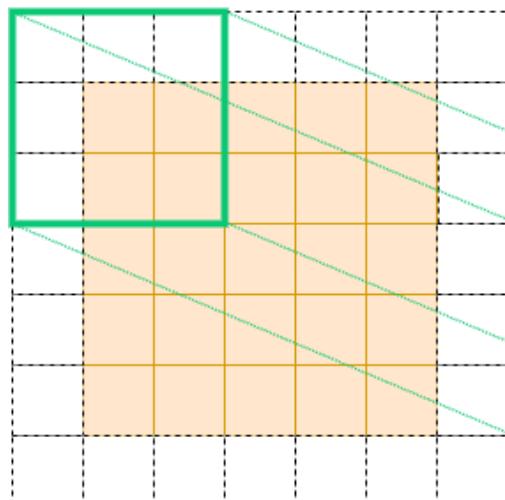
Filter N x N: 3 x 3

Output: 3 x 3



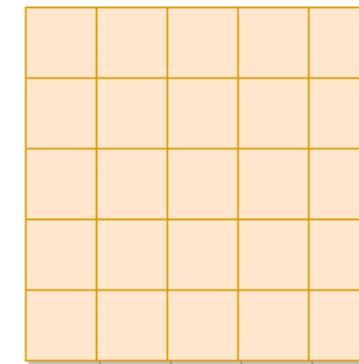
Input D x D: 5 x 5  
Plus added padding of size 1

Padding SAME  
Output dimension = Input dimension



Filter N x N: 3 x 3

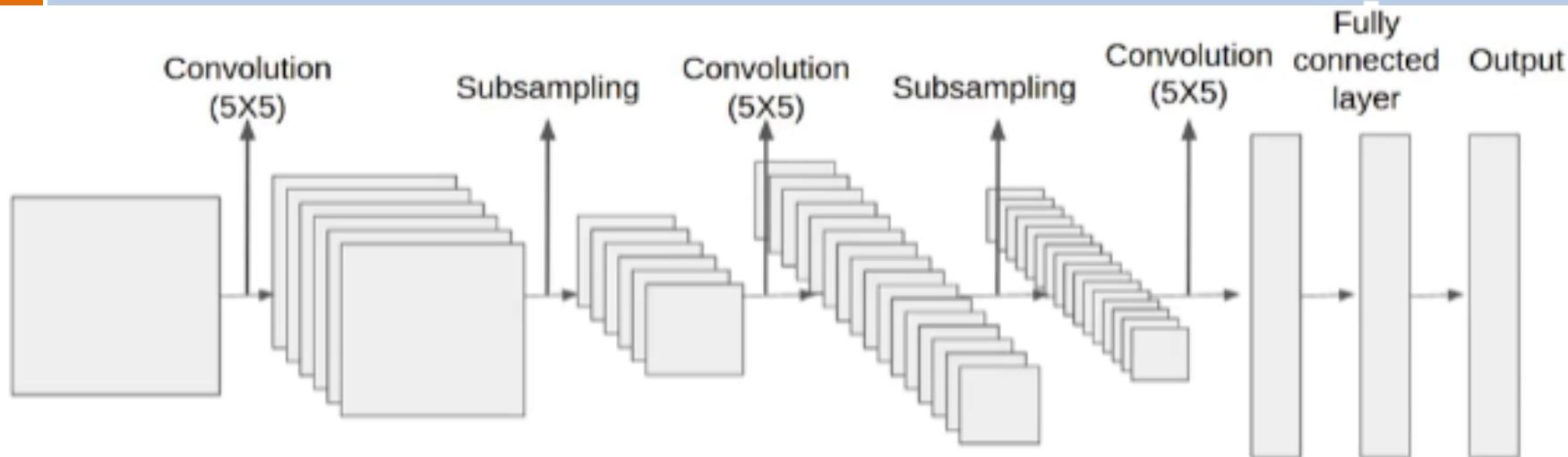
Output: 5 x 5



AI Geek Programmer, 2019,  
Convolutional neural  
network 2: architecture,  
<https://aigeekprogrammer.com/convolutional-neural-network-image-recognition-part-2/>

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이



Input	Feature Map 28 X 28 X 6	Feature Map 14 X 14 X 6	Feature Map 10 X 10 X 16	Feature Map 5 X 5 X 16	120	84	10
-------	----------------------------	----------------------------	-----------------------------	---------------------------	-----	----	----

Layer	# filters / neurons	Filter size	Stride	Size of feature map	Activation function
Input	-	-	-	32 X 32 X 1	
Conv 1	6	5 * 5	1	28 X 28 X 6	tanh
Avg. pooling 1		2 * 2	2	14 X 14 X 6	
Conv 2	16	5 * 5	1	10 X 10 X 16	tanh
Avg. pooling 2		2 * 2	2	5 X 5 X 16	
Conv 3	120	5 * 5	1	120	tanh
Fully Connected 1	-	-	-	84	tanh
Fully Connected 2	-	-	-	10	Softmax

$$O = \frac{I-F+2P}{S} + 1$$

$$O = \frac{I-F}{S} + 1$$

Shipra Saxena, 2021, The Architecture of Lenet-5, [analyticsvidhya.com/blog/2021/03/the-architecture-of-lenet-5/](http://www.analyticsvidhya.com/blog/2021/03/the-architecture-of-lenet-5/)

# 딥러닝의 구조적 이해

## CNN/RNN/LSTM의 개념적 차이

```
#Instantiate an empty model
model = Sequential()

# C1 Convolutional Layer
model.add(Conv2D(6, kernel_size=(5, 5), strides=(1, 1), activation='tanh', input_shape=input_shape, padding='same'))

# S2 Pooling Layer
model.add(AveragePooling2D(pool_size=(2, 2), strides=2, padding='valid'))

# C3 Convolutional Layer
model.add(Conv2D(16, kernel_size=(5, 5), strides=(1, 1), activation='tanh', padding='valid'))

# S4 Pooling Layer
model.add(AveragePooling2D(pool_size=(2, 2), strides=2, padding='valid'))

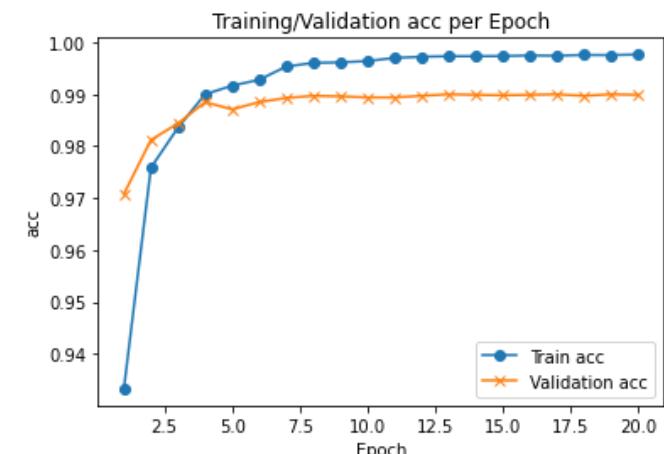
# C5 Fully Connected Convolutional Layer
model.add(Flatten())
model.add(Dense(120, kernel_size=(5, 5), strides=(1, 1), activation='tanh', padding='valid'))

#Flatten the CNN output so that we can connect it with fully connected layers
model.add(Flatten())

# FC6 Fully Connected Layer
model.add(Dense(84, activation='tanh'))

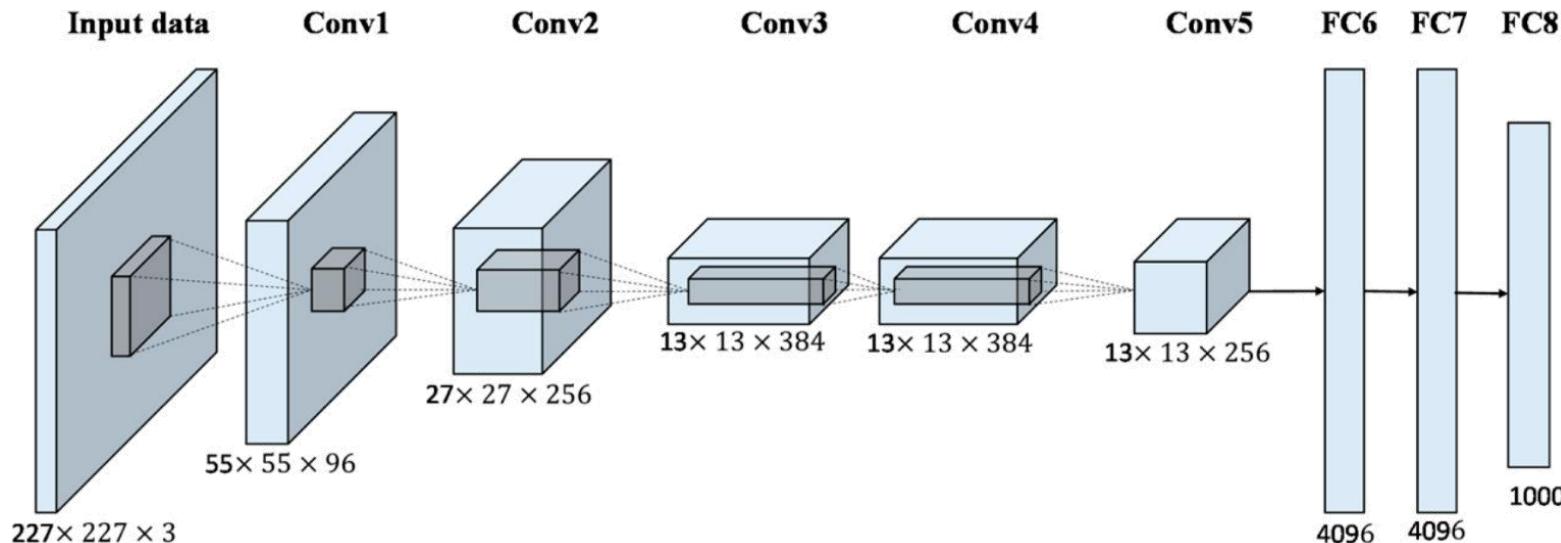
# Output Layer with softmax activation
model.add(Dense(10, activation='softmax'))

# print the model summary
model.summary()
```



# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이



```
model = Sequential(name="Alexnet")
```

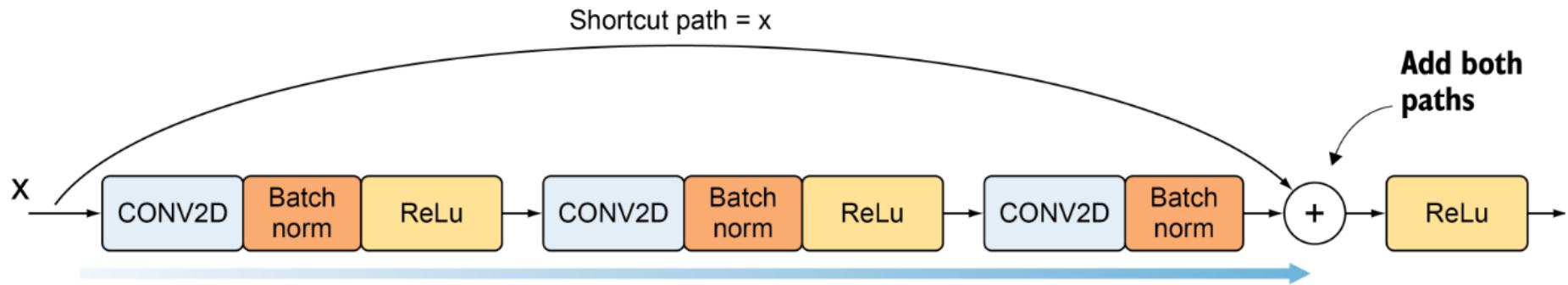
```
# 1st layer (conv + pool + batchnorm)
model.add(Conv2D(filters= 96, kernel_size= (11,11), strides=(4,4), padding='valid', kernel_regularizer=l2(0.0005),
input_shape = (227,227,3)))
model.add(Activation('relu'))
model.add(MaxPool2D(pool_size=(3,3), strides= (2,2), padding='valid'))
model.add(BatchNormalization())
```

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

## Residual Connections (Skip Connections)

ResNet과 같은 아키텍처에서 사용되는 방법으로, 특정 층의 입력이 그 층의 출력에 바로 합산



$X_{\text{shortcut}} = X$

$F = 32$

$X = \text{Conv2D}(\text{filters} = F, \text{kernel\_size} = (3, 3), \text{strides} = (1,1))(X)$

$X = \text{Activation('relu')}(X)$

$X = \text{Conv2D}(\text{filters} = F, \text{kernel\_size} = (3, 3), \text{strides} = (1,1))(X)$

$X = \text{Add()}([X, X_{\text{shortcut}}])$

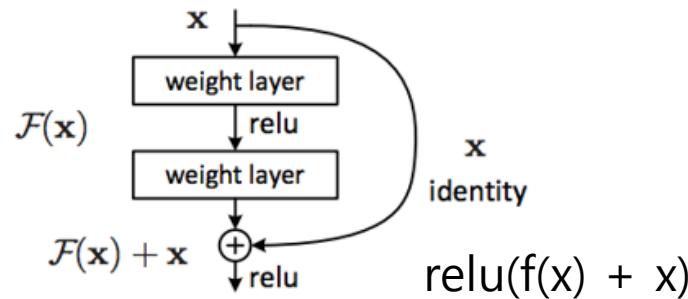
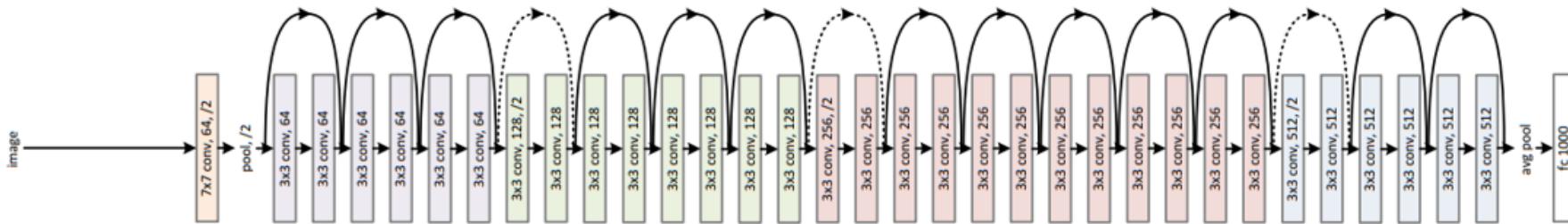
$X = \text{Activation('relu')}(X)$

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

## Residual Connections (Skip Connections)

### Residual Networks (ResNet50)

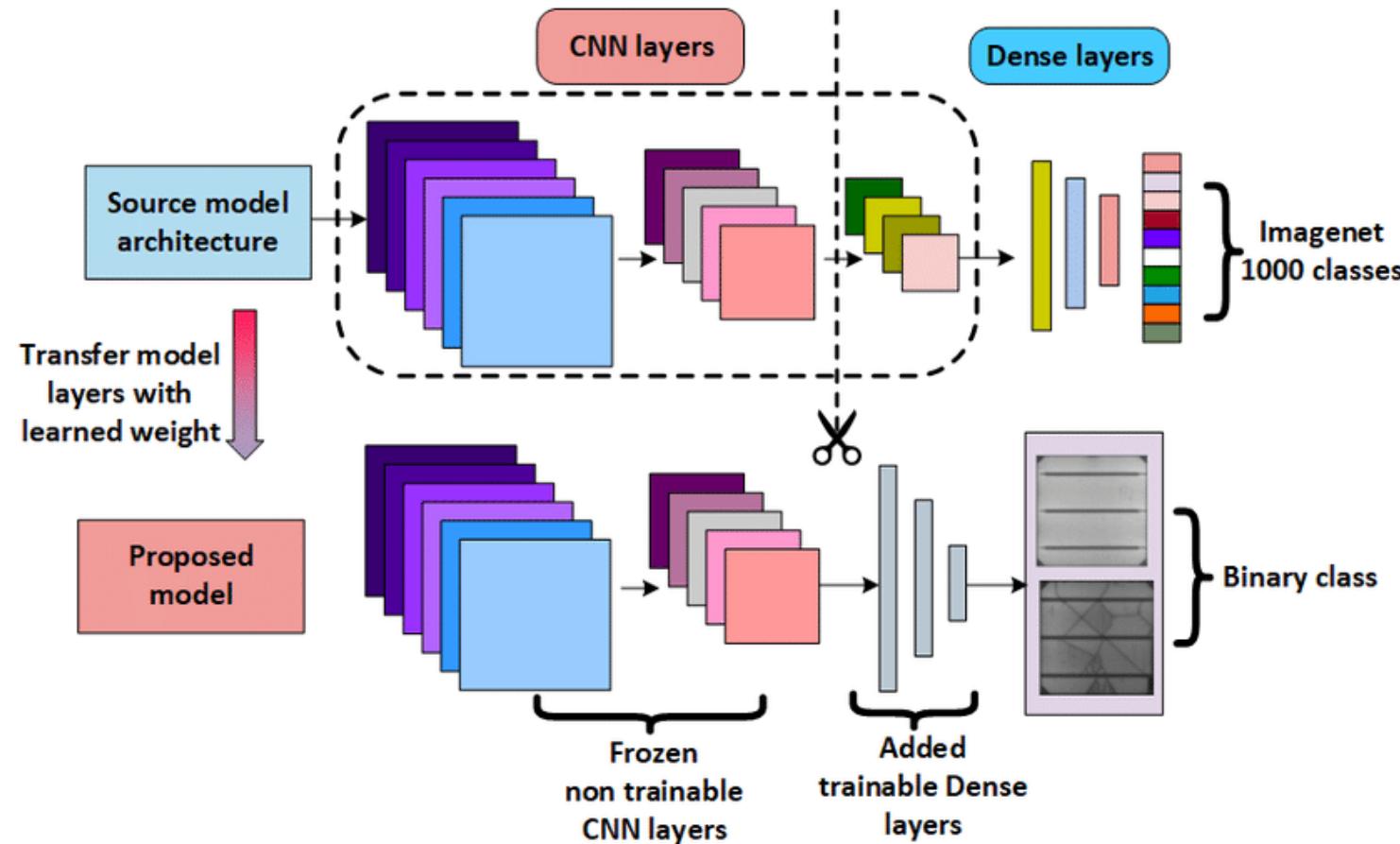


# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

## Transfer learning

미리 학습된(pre-trained) 모델의 지식과 특징을 새로운 관련 작업에 재사용하는 머신러닝 기법

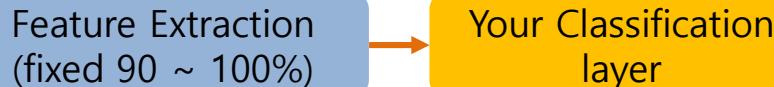


# 딥러닝의 구조적 이해

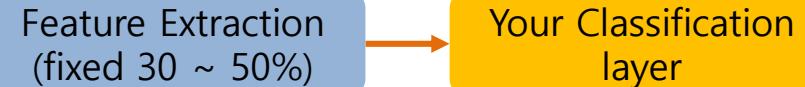
CNN/RNN/LSTM의 개념적 차이

## Transfer learning

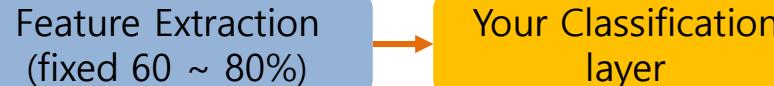
SS. Small target dataset size, Similar domains



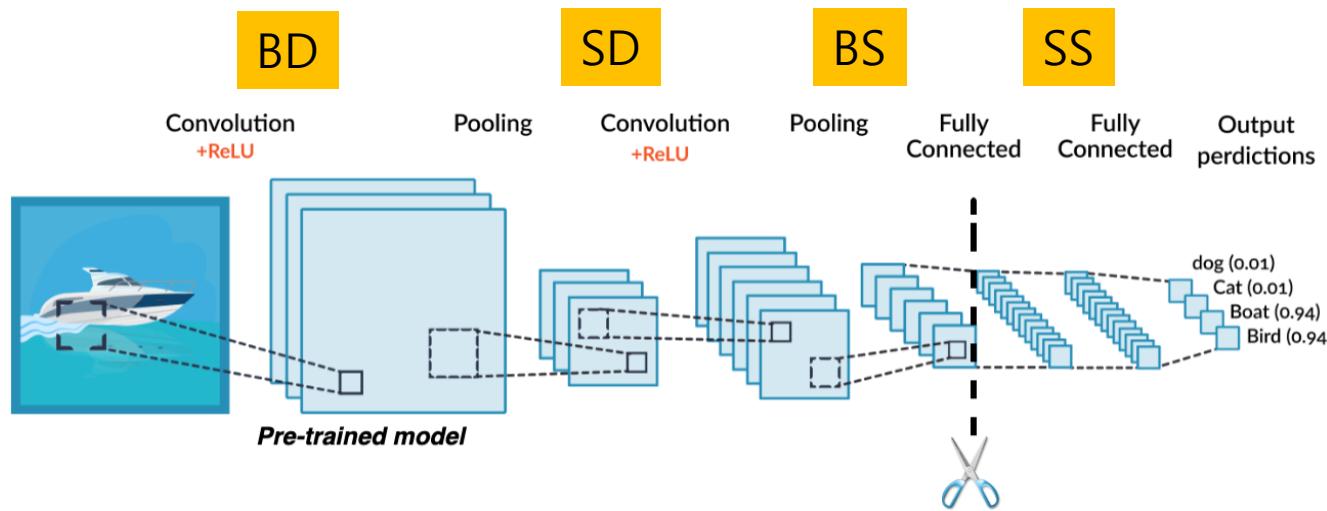
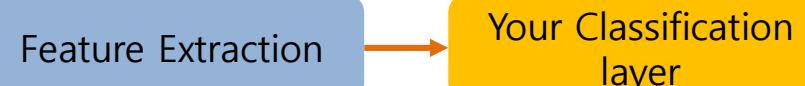
SD. Small target dataset size, Different domains



BS. Big target dataset size, Similar domains



BD. Big target dataset size, Different domains



# 딥러닝의 구조적 이해

Tensorflow vs Pytorch 비교

## 딥러닝 프레임워크 차이점

TensorFlow, PyTorch, Keras는 딥러닝 모델의 기울기(gradient)를 계산하고 업데이트하는 방식에서 구조적인 차이 있음. 특히, 편미분 연산자 정보 관리하는 자료구조(그래프 구상 방식)이 다름

딥러닝 모델은 파라미터들을 최적화하기 위해 손실 함수에 대한 각 파라미터의 기울기를 계산해야함. 역전파 알고리즘의 효과적 실행을 위해, 딥러닝 프레임워크는 자동 미분(Automatic Differentiation) 기능을 내장. 자동 미분은 크게 두 가지 방식으로 구현

정적 그래프 (Static Graph): 모델 실행 전에 전체 계산 그래프를 미리 정의. Tensorflow에 적용

동적 그래프 (Dynamic Graph) : 코드가 실행될 때마다 계산 그래프를 즉시 생성. PyTorch에 적용

특징	정적 (TensorFlow 1.x 기준, Keras)	동적 (PyTorch)
그래프 방식	모델 실행 전에 전체 계산 그래프를 미리 정의	코드가 실행될 때마다 계산을 수행, 동시에 그래프를 생성
장점	한 번 컴파일된 그래프는 최적화되어 빠른 실행 속도 제공. 분산 환경 학습 및 모바일/임베디드 장치 배포에 유리	일반 파이썬 코드처럼 중간 변수 값을 확인, 디버깅이 용이. 모델 구조가 동적으로 변하는 경우(가변 길이 시퀀스 등)에 적합
단점	그래프가 실행되기 전까지는 변수 값 디버깅 확인 어려움. 동적인 모델 구조 변화에 취약	정적 그래프에 비해 실행 시 추가 오버헤드 발생
기울기 연산자 연결 방식	tf.Graph 내에 연산 노드들이 연결되어 있으며, tf.GradientTape가 연산을 기록하여 기울기를 계산	torch.Tensor의 grad_fn 속성을 통해 연산들이 연결되어 역전파를 위한 계산 그래프를 형성

# 딥러닝의 구조적 이해

## Tensorflow vs Pytorch 비교



# 딥러닝의 구조적 이해

## Tensorflow vs Pytorch 비교



Get Started   Ecosystem   Mobile   Blog   **Tutorials**   Docs ▾   Resources ▾   GitHub

1.12.1+cu102

Search Tutorials

PyTorch Recipes [ + ]

Introduction to PyTorch [ - ]

Learn the Basics

● Quickstart

Tensors

Datasets & DataLoaders

Transforms

Build the Neural Network

Automatic Differentiation with `torch.autograd`

Optimizing Model Parameters

Save and Load the Model

Introduction to PyTorch on YouTube [ - ]

Introduction to PyTorch - YouTube Series

Introduction to PyTorch

Introduction to PyTorch Tensors

Tutorials > Quickstart



Run in Microsoft Learn

Run in Google Colab

Download Notebook

View on GitHub

[Learn the Basics](#) || **Quickstart** || [Tensors](#) || [Datasets & DataLoaders](#) || [Transforms](#) || [Build Model](#) || [Autograd](#) || [Optimization](#) || [Save & Load Model](#)

## QUICKSTART

This section runs through the API for common tasks in machine learning. Refer to the links in each section to dive deeper.

### Working with data

PyTorch has two **primitives to work with data**: `torch.utils.data.DataLoader` and `torch.utils.data.Dataset`. **Dataset stores the samples and their corresponding labels**, and `DataLoader` **wraps an iterable around the Dataset**.

```
import torch
from torch import nn
from torch.utils.data import DataLoader
from torchvision import datasets
from torchvision.transforms import ToTensor
```

# 딥러닝의 구조적 이해

## Tensorflow vs PyTorch 비교

### Tensorflow

```
import tensorflow as tf

x_data = [1, 2, 3, 4, 5]
y_data = [2, 4, 5, 4, 5]

model = tf.keras.Sequential([
    tf.keras.layers.Dense(1, input_dim=1)
])

model.compile(optimizer='sgd',
              loss='mean_squared_error')

model.fit(x_data, y_data, epochs=1000,
          verbose=0)

print(model.predict([10]))
```

### PyTorch

```
import torch, torch.nn as nn, torch.optim as optim

x_data = torch.tensor([[1.], [2.], [3.], [4.], [5.]])
y_data = torch.tensor([[2.], [4.], [5.], [4.], [5.]])
```

```
class LinearRegressionModel(nn.Module):
    def __init__(self):
        super(LinearRegressionModel, self).__init__()
        self.linear = nn.Linear(1, 1)

    def forward(self, x):
        return self.linear(x)
```

```
model = LinearRegressionModel()
optimizer = optim.SGD(model.parameters(), lr=0.01)

for epoch in range(1000):
    optimizer.zero_grad()
    outputs = model(x_data)
    loss = criterion(outputs, y_data)
    loss.backward()
    optimizer.step()

print(model(torch.tensor([[10.]])))
```

# 딥러닝의 구조적 이해

## Pytorch vs Keras 비교

Keras

```
from keras.models import Sequential  
from keras.layers import Dense, Dropout,  
Conv2D, MaxPool2D, Flatten  
from keras.layers import  
BatchNormalization  
  
model = Sequential()  
model.add(Conv2D(32, (3, 3),  
activation='relu', input_shape=(32, 32, 3)))  
model.add(MaxPool2D())  
model.add(Conv2D(16, (3, 3),  
activation='relu'))  
model.add(MaxPool2D())  
model.add(Flatten())  
model.add(Dense(10, activation='softmax'))  
  
model.summary()
```

PyTorch

```
import torch  
  
nn = torch.nn  
F = torch.nn.functional  
  
class Net(nn.Module):  
    def __init__(self):  
        super(Net, self).__init__()  
        self.conv1 = nn.Conv2d(3, 32, 3)  
        self.conv2 = nn.Conv2d(32, 16, 3)  
        self.fc1 = nn.Linear(16 * 6 * 6, 10)  
        self.pool = nn.MaxPool2d(2, 2)  
  
    def forward(self, x):  
        x = self.pool(F.relu(self.conv1(x)))  
        x = self.pool(F.relu(self.conv2(x)))  
        x = x.view(-1, 16 * 6 * 6)  
        x = F.log_softmax(self.fc1(x), dim=-1)  
        return x  
  
model = Net()  
print(model)
```

# 딥러닝의 구조적 이해

Tensorflow vs Pytorch 비교

1 Load data

2 Define model

3 Train model

4 Evaluate model

## General

PyTorch is an open source machine learning framework. It uses `torch.Tensor` – multi-dimensional matrices – to process. A core feature of neural networks in PyTorch is the autograd package, which provides automatic derivative calculations for all operations on tensors.

```
import torch  
import torch.nn as nn  
from torchvision import datasets, models, transforms  
import torch.nn.functional as F
```

Root package  
Neural networks  
Popular image datasets, architectures & transforms  
Collection of layers, activations & more

<code>torch.randn(*size)</code>	Create random tensor
<code>torch.Tensor(L)</code>	Create tensor from list
<code>tnsr.view(a,b, ...)</code>	Reshape tensor to size (a, b, ...)
<code>requires_grad=True</code>	tracks computation history for derivative calculations

## Layers

 `nn.Linear(m, n)`: Fully Connected layer (or dense layer) from m to n neurons

 `nn.Flatten()`: Flattens a contiguous range of dimensions into a tensor

 `nn.Dropout(p=0.5)`: Randomly sets input elements to zero during training to prevent overfitting

 `nn.Embedding(m, n)`: Lookup table to map dictionary of size m to embedding vector of size n

 `nn.ConvXd(m, n, s)`: X-dimensional convolutional layer from m to n channels with kernel size s;  $X \in \{1, 2, 3\}$

 `nn.MaxPoolXd(s)`: X-dimensional pooling layer with kernel size s;  $X \in \{1, 2, 3\}$

 `nn.BatchNormXd(n)`: Normalizes a X-dimensional input batch with n features;  $X \in \{1, 2, 3\}$

 `nn.RNN/LSTM/GRU`: Recurrent networks connect neurons of one layer with neurons of the same or a previous layer

[PyTorch Tutorial for Reshape, Squeeze, Unsqueeze, Flatten and View - MLK - Machine Learning Knowledge](#)

[Link - <https://www.stefanseegerer.de/media/pytorch-cheatsheet-EN.pdf>](#)

`torch.nn` offers a bunch of other building blocks.

A list of state-of-the-art architectures can be found at <https://paperswithcode.com/sota>.

# 자연어 처리

텍스트 전처리  
토큰과 단어 임베딩  
Hugging Face 모델 예시 소개

# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

딥러닝 모델은 텍스트를 바로 이해하지 못함

문장 → 토크나이징(tokenizing) → 토큰 ID → 임베딩 벡터

"I am a student" → ["I", "am", "a", "student"]

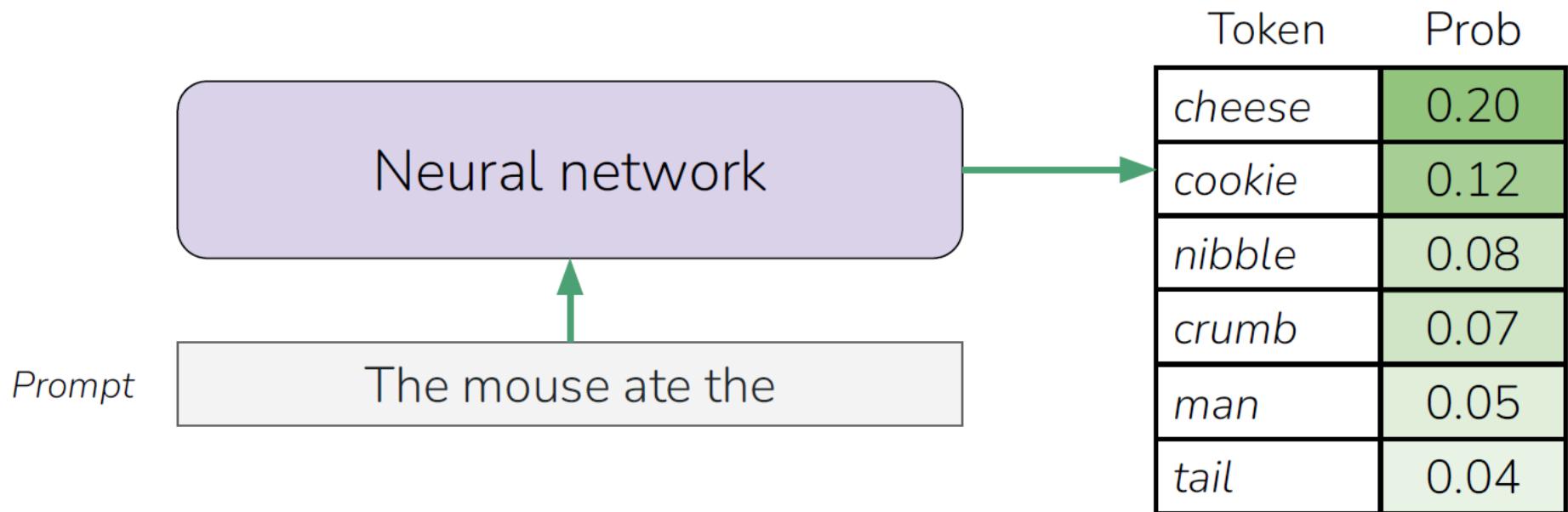
["I", "am", "a", "student"] → [101, 2023, 1037, 3076]

embedding = nn.Embedding(vocab\_size, d\_model)  
token\_vec = embedding(token\_ids)

[-0.1, 0.3, ..., 0.05]

# 자연어 처리의 기본

텍스트 전처리 (토크나이징, 정제)



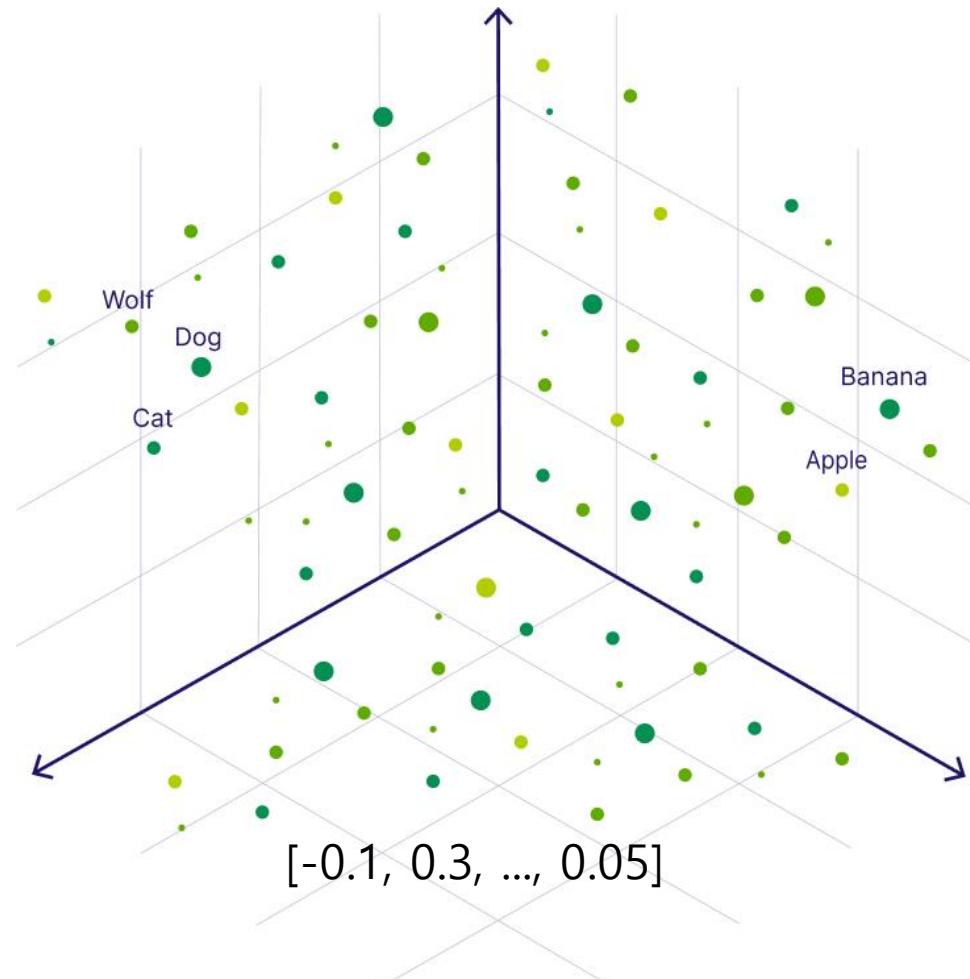
# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

## 임베딩 벡터

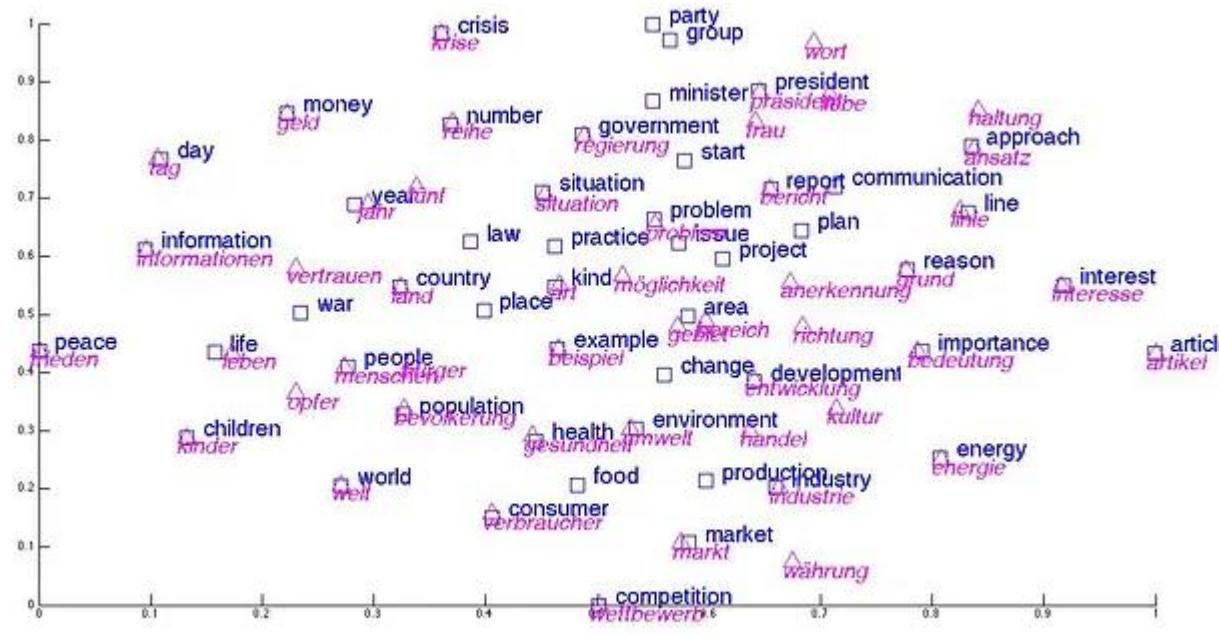
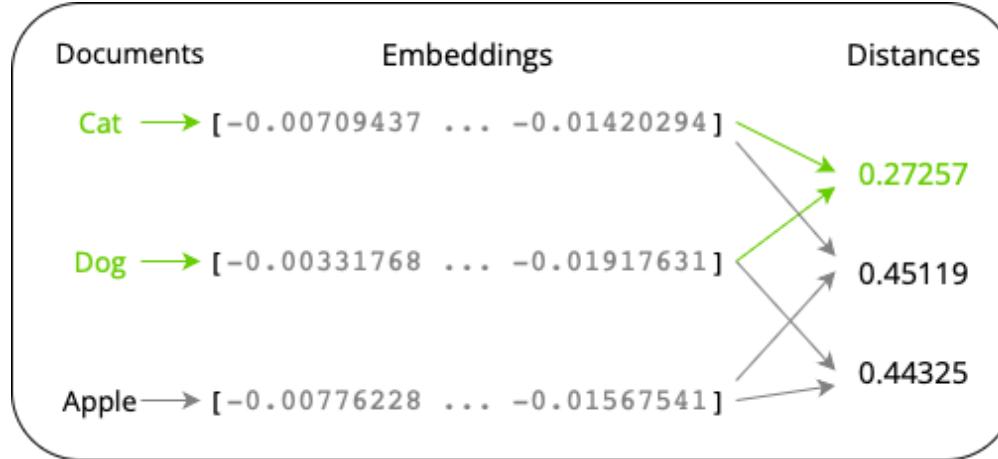
단어, 문장, 이미지, 사용자 등 고차원적이고 희소한(sparse) 데이터를 저차원, 연속적인(dense) 벡터 공간으로 압축하여 표현한 것

토큰 사전과 임베딩 모델이 다르면, 제대로 된 모델 학습, 예측, 패턴 계산 결과를 얻기 어려움



# 자연어 처리의 기본 단어 임베딩 (Word2Vec)

## 임베딩 벡터

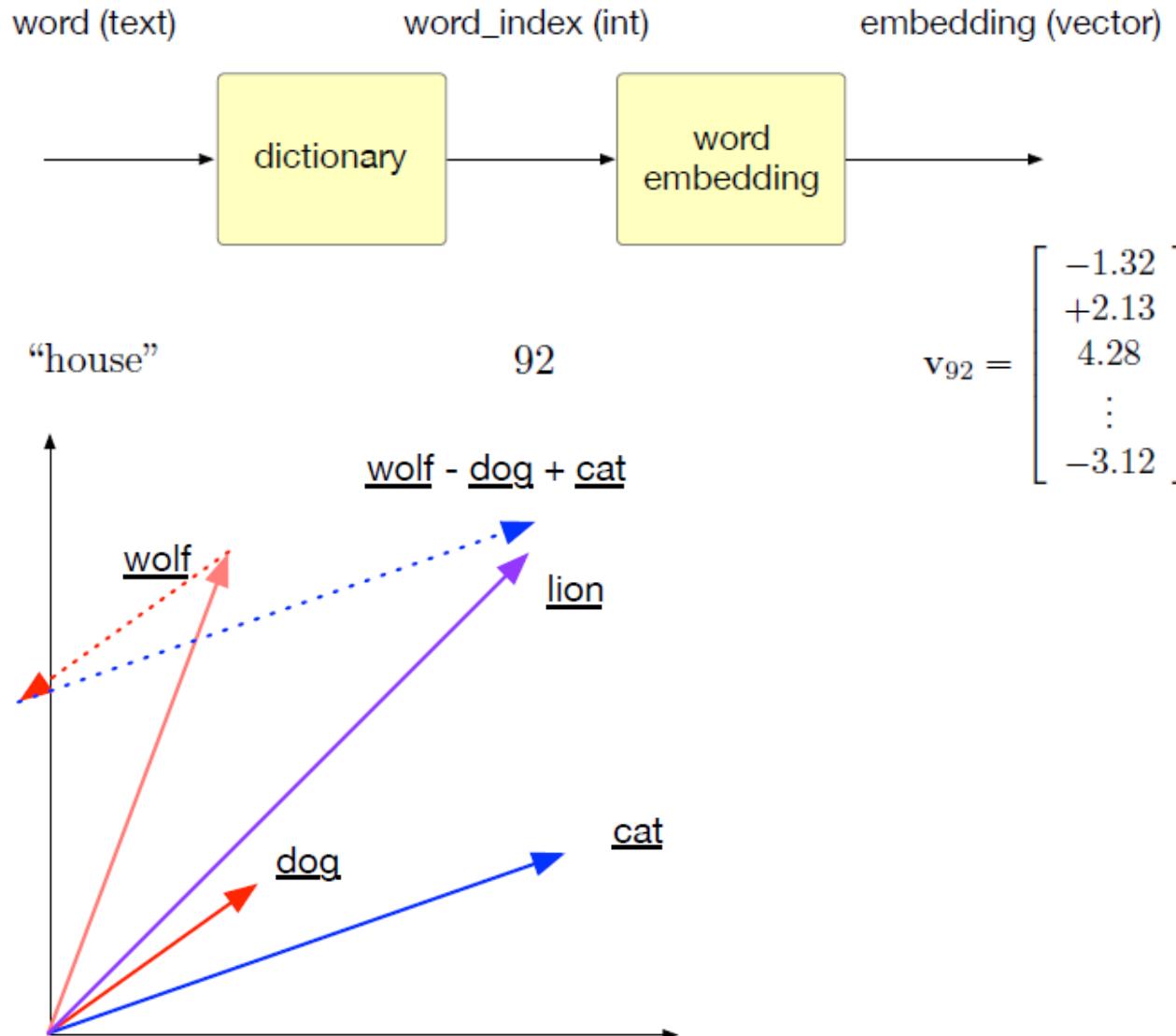


<https://daddynkidsmakers.blogspot.com/2023/12/blog-post.html>

# 자연어 처리의 기본

텍스트 전처리 (토크나이징, 정제)

## 임베딩 벡터



# 자연어 처리의 기본

## 단어 임베딩 (Word2Vec)

### Word2Vec

구글이 2013년에 제안한 임베딩 방법으로, Skip-gram과 CBOW(Continuous Bag of Words) 두 가지 학습 구조 사용

Skip-gram은 중심 단어를 입력으로 주고 주변 단어를 예측하는 방식. CBOW는 주변단어들로 중심단어를 예측하는 방식

"The quick brown fox jumps" 문장 중심 단어 " fox"의 윈도 크기가 2라면, 주변 단어는 "brown" 과 "jumps" 가 됨

Word2Vec은 주변 단어를 예측하는 loss를 최소화하며 학습됨. 임베딩 모델들은 주로 대규모 텍스트 데이터에서 비지도 학습 방식으로 학습

```
# Word2Vec Skip-gram 의사코드
for center_word, context_words in dataset:
    center_vec = word_embedding(center_word)
    for context in context_words:
        context_vec = word_embedding(context)
        loss += negative_sampling_loss(center_vec, context_vec)
    loss.backward()
    optimizer.step()
```

# 자연어 처리의 기본

텍스트 전처리 (토크나이징, 정제)

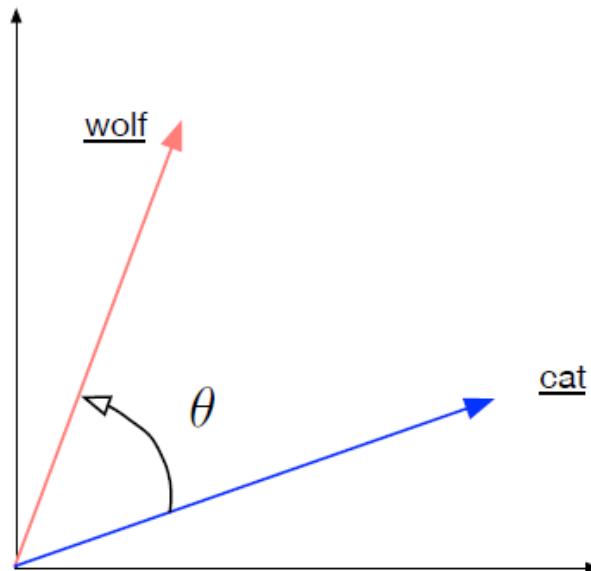
## 코사인 유사도(Cosine Similarity)

두 벡터의 방향이 얼마나 유사한지를 측정하는 척도

워드 임베딩에서 “고양이”과 “표범”처럼 의미가 유사한 단어들은 벡터 공간에서 서로 가까운 방향을 가리키도록 학습되는데, 이때 이들의 유사성을 코사인 유사도로 측정 가능

$$\cos(\theta) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|}$$

- 1: 두 벡터가 완전히 같은 방향을 가리킬 때 (가장 유사함)
- 0: 두 벡터가 직교(수직)할 때
- 1: 두 벡터가 완전히 반대 방향을 가리킬 때



# 자연어 처리의 기본

정규식

## 정규표현식(Regular Expression)

문자열에서 특정 패턴을 찾거나, 치환하거나, 분리하기 위한 문자열 검색 도구

1950년대 미국 수학자 [스티븐 클리\(Stephen Kleene\)](#)에 의해 처음 소개

문법	설명	예시	설명
.	임의의 한 문자	a.c	'abc', 'axc'에 매칭
^	문자열의 시작	^a	'apple'에 매칭, 'banana'에는 X
\$	문자열의 끝	e\$	'apple', 'title'에 매칭
[]	문자 집합	[aeiou]	모음에 매칭
[^ ]	제외 문자 집합	[^0-9]	숫자가 아닌 문자
*	0 개 이상의 반복	a*	", 'a', 'aa' 모두 매칭
+	1 개 이상의 반복	a+	'a', 'aa'는 매칭, "은 X
?	0 또는 1 개	a?	", 'a'는 매칭
{n}	정확히 n번 반복	a{3}	'aaa'에만 매칭
{n,}	n번 이상 반복	a{2,}	'aa', 'aaa', ...에 매칭
{n,m}	n~m번 반복	a{1,3}	'a', 'aa', 'aaa'에 매칭
₩d	숫자 문자	₩d+	'123', '5' 등에 매칭
₩w	단어 문자	₩w+	'abc123', '_' 등에 매칭
₩s	공백 문자	₩s+	' ', '₩t', '₩n' 등에 매칭
0	그룹화	(abc)+	'abc', 'abcaabc'에 매칭

# 자연어 처리의 기본

## 텍스트 전처리 (토크나이저)

### WordPiece

토큰화 프로세스를 사용하여 하위 단어 단위로 토큰화

### BPE(Byte Pair Encoding)

서브워드 토크나이저 한 종류로 데이터의 가장 일반적인 연속 바이트 쌍을 해당 데이터 내에 발생하지 않는 바이트로 대체하는 데이터 압축함

BPE는 초기 텍스트 압축 알고리즘으로 개발된 후 GPT 모델 사전 학습 시 토큰화를 위해 사용

단어를 보다 관리하기 쉬운 하위 단어나 기호로 분할하여 대규모 어휘를 효율적으로 인코딩

어휘 크기를 크게 줄이고 희귀 단어나 OOV(어휘에서 벗어난) 용어를 처리하는 모델 능력 향상

```
tokenizer.encode('123234234')
[10163, 24409, 24409]
tokenizer.encode('123')
[10163]
tokenizer.encode('234')
[24409]
tokenizer.encode('234')
[24409]
```

# 자연어 처리의 기본

## 텍스트 전처리 (토크나이저)

```
from transformers import DistilBertTokenizerFast, DistilBertModel
```

```
tokenizer = DistilBertTokenizerFast.from_pretrained("distilbert-base-uncased")
tokens = tokenizer.encode('This is a IfcBuilding.', return_tensors='pt')
print("These are tokens!", tokens)
for token in tokens[0]:
    print("This are decoded tokens!", tokenizer.decode([token]))
```

```
model = DistilBertModel.from_pretrained("distilbert-base-uncased")
print(model.embeddings.word_embeddings(tokens))
for e in model.embeddings.word_embeddings(tokens)[0]:
    print("This is an embedding!", e)
```

```
These are tokens! tensor([[ 101,  2023,  2003,  1037,  2065, 27421, 19231,  4667,  1012,   102]])
This are decoded tokens! [CLS]
This are decoded tokens! this
This are decoded tokens! is
This are decoded tokens! a
This are decoded tokens! if
This are decoded tokens! ##cb
This are decoded tokens! ##uil
This are decoded tokens! ##ding
This are decoded tokens! .
This are decoded tokens! [SEP]
```

```
This is an embedding! tensor([ 3.8952e-02, -1.2318e-02, -2.0844e-02, -5.2684e-04, -1.9758e-02,
  3.8324e-02, -2.0617e-02,  3.3877e-03, -2.2452e-02, -4.3990e-02,
  1.2990e-02, -1.6670e-02,  9.7562e-03, -1.2669e-02, -4.5170e-02,
  2.5090e-02,  4.4694e-02,  5.9726e-02, -5.0432e-03, -3.6367e-02,
```

# 자연어 처리의 기본

## 텍스트 전처리 (토크나이저)

```
from transformers import BertTokenizerFast, BertModel
import torch
from torch import nn

# BERT 토크나이저 사전학습모델 로딩
tokenizer = BertTokenizerFast.from_pretrained('bert-base-uncased')
print(tokenizer.tokenize("[CLS] Hello world, how are you?"))

print(tokenizer.tokenize("[newtoken] Hello world, how are you?"))
tokenizer.add_tokens(['[newtoken]'])

['[',
 'newt',
 '#oke',
 '#n',
 ']',
 'hello',
 'world',
 '',
 '',
 'how',
 'are',
 'you',
 '?']
```

# 자연어 처리의 기본

텍스트 전처리 (토크나이저)

# 토큰을 추가하고 다시 토큰화를 한다.

```
tokenizer.add_tokens(['[newtoken]'])
```

```
tokenizer.tokenize("[newtoken] Hello world, how are you?")
```

# 제대로 토큰화가 된다.

```
['[newtoken]', 'hello', 'world', ',', 'how', 'are', 'you', '?']
```

# 토큰값을 확인해 본다.

```
tokenized = tokenizer("[newtoken] Hello world, how are you?", add_special_tokens=False,  
return_tensors="pt")  
print(tokenized['input_ids'])
```

```
tkn = tokenized['input_ids'][0, 0]
```

```
print("First token:", tkn)
```

```
print("Decoded:", tokenizer.decode(tkn))
```

# 다음과 같이, 토큰값이 잘 할당된 것을 알 수 있다.

```
tensor([[30522, 7592, 2088, 1010, 2129, 2024, 2017, 1029]])
```

```
First token: tensor(30522)
```

```
Decoded: [newtoken]
```

# 자연어 처리의 기본

Named Entity Recognition (NER)

## 개체명 인식(Named Entity Recognition, NER)

문장에서 특정한 개체를 찾아내고, 이를 사람(Person), 조직(Organization), 장소(Location) 등의 범주로 분류하는 기술. 감정 분석(sentiment analysis)이나 뉴스 기사 분류(news classification) 등 사용 가능

"Apple Inc. was founded by Steve Jobs"라는 문장에서 "Apple Inc."는 조직, "Steve Jobs"는 사람으로 분류

B- (Begin): 개체명의 시작

B-PER: 사람 이름의 시작

B-ORG: 조직명의 시작

B-LOC: 장소명의 시작

I- (Inside): 개체명의 내부/연속

I-PER: 사람 이름의 연속 부분

I-ORG: 조직명의 연속 부분

I-LOC: 장소명의 연속 부분

## N-gram

연속된 N개의 단어 또는 문자 단위 묶음을 의미. 자연어 처리(NLP)에서 주로 문맥 모델링에 사용

Unigram (1-gram): 단어 단위

[("나는", "밥을", "먹었다")]

Bigram (2-gram): 연속된 2개 단어

[("나는", "밥을"), ("밥을", "먹었다")]

Trigram (3-gram): 연속된 3개 단어

[("나는", "밥을", "먹었다")]

# 자연어 처리의 기본

BLEU

## BLEU(Bilingual Evaluation Understudy)

기계 번역 결과를 평가하는 지표. 번역문장(후보)와 기준문장 간의 n-gram 일치 정도를 계산

$$BLEU = BP \cdot \exp \left( \sum_{n=1}^N w_n \log p_n \right)$$

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

BP는 번역이 기준 번역보다 짧을 경우 부여하는 패널티 (Brevity Penalty).  $w_n$ 은 각 n-gram 별 가중치( $N=4$ ,  $w_n = 0.25$ )

기준문장 "the cat is on the mat", 기계번역 "the cat is on mat"라면, 1-gram 정밀도는 5개 중 4개가 일치하므로 0.8

후보 문장: "the dog is on the mat"

참조 문장: "the cat is on the mat"

후보 문장 1-gram: "the", "dog", "is", "on", "the", "mat"

참조 문장 1-gram: "the", "cat", "is", "on", "the", "mat"

각 1-gram 일치 횟수:

"the": 후보 2개, 참조 최대 2개  $\rightarrow \min(2,2)=2$

"dog": 후보 1개, 참조 최대 0개  $\rightarrow \min(1,0)=0$

"is": 후보 1개, 참조 최대 1개  $\rightarrow \min(1,1)=1$

"on": 후보 1개, 참조 최대 1개  $\rightarrow \min(1,1)=1$

"mat": 후보 1개, 참조 최대 1개  $\rightarrow \min(1,1)=1$

"cat": 후보 0개, 참조 최대 1개 (참조에만 있음)

일치하는 1-그램의 합 =  $2+0+1+1+1=5$

후보 문장 총 1-그램 수 = 6

$P_1=5/6 \approx 0.8333$

후보 문장 2-gram: "the dog", "dog is", "is on", "on the", "the mat"

참조 문장의 2-gram: "the cat", "cat is", "is on", "on the", "the mat"

각 2-그램의 일치 횟수:

"the dog":  $\min(1,0)=0$

"dog is":  $\min(1,0)=0$

"is on":  $\min(1,1)=1$

"on the":  $\min(1,1)=1$

"the mat":  $\min(1,1)=1$

일치하는 2-gram 합 =  $0+0+1+1+1=3$

후보 문장 총 2-gram 수 = 5.  $P_2=3/5=0.6$

$$BLEU = BP \cdot \exp (w_1 \log p_1 + w_2 \log p_2)$$

$$BLEU = 1.0 \cdot \exp (0.5 \cdot \log(0.8333) + 0.5 \cdot \log(0.6))$$

$$BLEU \approx 0.7071$$

# 자연어 처리의 기본

Hugging Face 모델 예시 소개

## 허깅페이스(Hugging Face)

최신 딥러닝 모델들을 쉽게 사용할 수 있도록 표준화된 인터페이스를 제공하는 것으로 유명. 우수한 LLM, 트랜스포머 라이브러리, 학습 데이터셋, 모델 실행 스페이스 기능 제공

The screenshot shows the Hugging Face platform interface. At the top, there's a navigation bar with a logo, a search bar, and links for Models, Datasets, Spaces, Community, Docs, Enterprise, and Pricing. Below the search bar, there are buttons for Papers and arxiv:1810.04805. The main content area displays a model card for 'google-bert/bert-base-uncased'. The card includes the model name, a thumbnail, a summary, and a detailed description. It also features sections for 'Upvote' (19), 'Models citing this paper' (293), and 'Datasets citing this paper' (4). Each section lists specific models or datasets along with their details like size, update date, and download count.

Models · Datasets · Spaces · Community · Docs · Enterprise · Pricing

Papers arxiv:1810.04805

## google-bert/bert-base-uncased · Hugging Face

### BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Published on Oct 11, 2018

Authors: Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova

#### Abstract

BERT is a bidirectional transformer-based model that pre-trains on unlabeled text and fine-tunes for various NLP tasks, achieving state-of-the-art results across multiple benchmarks.

AI-generated summary

We introduce a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models, BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications. BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

Upvote 19

Models citing this paper 293

google-bert/bert-base-uncased  
Fill-Mask · 0.1B · Updated Feb 19, 2024 · 59M · 2.35k

google-bert/bert-base-chinese  
Fill-Mask · 0.1B · Updated 20 days ago · 3.35M · 1.27k

dslim/bert-base-NER  
Token Classification · 0.1B · Updated Oct 8, 2024 · 2.25M · 623

google-bert/bert-base-multilingual-cased  
Fill-Mask · 0.2B · Updated Feb 19, 2024 · 10.1M · 513

Browse 293 models citing this paper

Datasets citing this paper 4

Babelscape/wikineurual  
Viewer · Updated Nov 13, 2022 · 1.03M · 3.31k · 33

# 자연어 처리의 기본 Hugging Face 모델 예시 소개

Hugging Face Search models, datasets, users...

Models 1,871,163 Filter by name

Full-text search Sort: Most likes

Main Tasks Libraries Languages Licenses Other

Tasks

- Text Generation Any-to-Any Image-Text-to-Text
- Image-to-Text Image-to-Image Text-to-Image
- Text-to-Video Text-to-Speech +42

Parameters

<1B 6B 12B 32B 128B >500B

Libraries

- PyTorch TensorFlow JAX Transformers
- Diffusers Safetensors ONNX GGUF
- Transformers.js MLX MLX Keras +40

Apps

- vLLM TGI llama.cpp MLX LM
- LM Studio Ollama Jan +12

Models

- deepseek-ai/DeepSeek-R1 Text Generation · 685B · Updated Mar 27 · 844k · 12.5k
- CompVis/stable-diffusion-v1-4 Text-to-Image · Updated Aug 24, 2023 · 3.48M · 6.87k
- meta-llama/Meta-Llama-3-8B Text Generation · 8B · Updated Sep 28, 2024 · 314k · 6.25k
- stabilityai/stable-diffusion-3-medium Text-to-Image · Updated Aug 12, 2024 · 13.4k · 4.8k
- openai/whisper-large-v3 Automatic Speech Recognition · 2B · Updated A... · 3.05M · 4.65k
- mistralai/Mixtral-8x7B-Instruct-v0.1 Text Generation · 47B · Updated Aug 19, 2024 · 388k · 4.49k
- meta-llama/Llama-3.1-8B-Instruct Text Generation · 8B · Updated Sep 26, 2024 · 6.25M · 4.31k
- black-forest-labs/FLUX.1-dev Text-to-Image · Updated 20 days ago · 1.44M · 10.9k
- stabilityai/stable-diffusion-xl-base-1.0 Text-to-Image · Updated Oct 31, 2023 · 2.4M · 6.73k
- bigscience/bloom Text Generation · 176B · Updated Jul 29, 2023 · 3.82k · 4.92k
- hexgrad/Kokoro-82M Text-to-Speech · Updated Apr 11 · 1.5M · 4.67k
- meta-llama/Llama-2-7b-chat-hf Text Generation · 7B · Updated Apr 17, 2024 · 1.04M · 4.5k
- meta-llama/Llama-2-7b Text Generation · Updated Apr 17, 2024 · 4.37k
- black-forest-labs/FLUX.1-schnell Text-to-Image · Updated Aug 16, 2024 · 651k · 4.09k

# 자연어 처리의 기본 Hugging Face 모델 예시 소개

[google/gemma-2b-it · Hugging Face](https://huggingface.co/google/gemma-2b-it)

 You need to agree to share your contact information to access this model

This repository is publicly accessible, but you have to accept the conditions to access its files and content.

By agreeing you accept to share your contact information (email and username) with the repository authors.

Agree and access repository

Cancel



Text Generation



Transformers



Safetensors



Korean

llama



Inference Endpoints

 Model card

 Files and versions

 Community 1



Gated model You have been granted access to this model

# 자연어 처리의 기본 Hugging Face 모델 예시 소개

Hugging Face Search models, datasets, users...

Models Datasets Spaces Community Docs Enterprise Pricing

google/gemma-7b like 3.18k Follow Google 20.5k ✓

Text Generation Transformers Safetensors GGUF gemma text-generation-inference arxiv:24 papers License: gemma

Model card Files and versions Community 120 Edit model card

Gated model You have been granted access to this model

## Gemma Model Card

Model Page: Gemma

This model card corresponds to the 7B base version of the Gemma model. You can also visit the model card of the [2B base model](#), [7B instruct model](#), and [2B instruct model](#).

Resources and Technical Documentation:

- [Gemma Technical Report](#)
- [Responsible Generative AI Toolkit](#)
- [Gemma on Kaggle](#)
- [Gemma on Vertex Model Garden](#)

Downloads last month 73,221

Safetensors Model size 8.54B params Tensor type BF16 Files info

Inference Providers NEW

Text Generation This model isn't deployed by any Inference Provider. Ask for provider support

Model tree for google/gemma-7b

- Adapters 9182 models
- Finetunes 360 models
- Merges 7 models
- Quantizations 23 models

google/gemma-7b · Hugging Face

# 자연어 처리의 기본

## Hugging Face 모델 예시 소개

### Fine-tuning examples

You can find fine-tuning notebooks under the [examples/ directory](#). We provide:

- A script to perform Supervised Fine-Tuning (SFT) on UltraChat dataset using QLoRA
- A script to perform SFT using FSDP on TPU devices
- A notebook that you can run on a free-tier Google Colab instance to perform SFT on English quotes dataset. You can also find the copy of the notebook [here](#).

### Running the model on a CPU

```
from transformers import AutoTokenizer, AutoModelForCausalLM

tokenizer = AutoTokenizer.from_pretrained("google/gemma-7b")
model = AutoModelForCausalLM.from_pretrained("google/gemma-7b")

input_text = "Write me a poem about Machine Learning."
input_ids = tokenizer(input_text, return_tensors="pt")

outputs = model.generate(**input_ids)
print(tokenizer.decode(outputs[0]))
```

### Running the model on a single / multi GPU

```
# pip install accelerate
from transformers import AutoTokenizer, AutoModelForCausalLM

tokenizer = AutoTokenizer.from_pretrained("google/gemma-7b")
model = AutoModelForCausalLM.from_pretrained("google/gemma-7b", device_map="auto")

input_text = "Write me a poem about Machine Learning."
input_ids = tokenizer(input_text, return_tensors="pt").to("cuda")
```

### Quantized Versions through bitsandbytes

- Using 8-bit precision (int8)

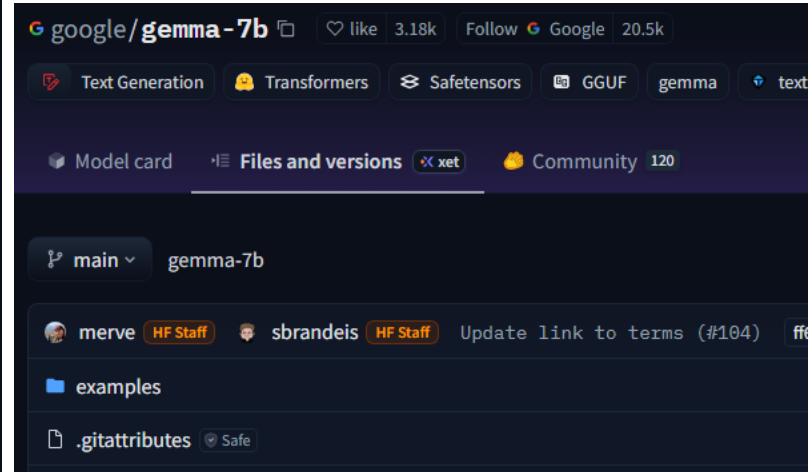
```
# pip install bitsandbytes accelerate
from transformers import AutoTokenizer, AutoModelForCausalLM

quantization_config = BitsAndBytesConfig(load_in_8bit=True)

tokenizer = AutoTokenizer.from_pretrained("google/gemma-7b")
model = AutoModelForCausalLM.from_pretrained("google/gemma-7b", quantization_config=quantization_config)

input_text = "Write me a poem about Machine Learning."
input_ids = tokenizer(input_text, return_tensors="pt").to("cuda")

outputs = model.generate(**input_ids)
print(tokenizer.decode(outputs[0]))
```



# 자연어 처리의 기본 Hugging Face 모델 예시 소개

Datasets: sanjay920/gemma-function-calling like 8

Split (1)  
train · 112k rows

Dataset

Search this dataset

messages  
list · lengths

3+6 67.9%

```
[ { "content": "You are a helpful assistant, with no  
access to external functions.", "role": "system" }, {..}
```

```
[  
{  
  "content": "You are a helpful assistant with access  
to the following functions. Use them if required -  
\n[\\n {\\n \"type\": \"function\",\\n \"function\":  
{\\n \"name\": \"send_email\",\\n \"description\":  
\"Send an email to a recipient\",\\n \"parameters\":  
{\\n \"type\": \"object\",\\n \"properties\": {\\n  
\"recipient\": {\\n \"type\": \"string\",\\n  
\"description\": \"The email address of the  
recipient\"\\n },\\n \"subject\": {\\n \"type\":  
\"string\",\\n \"description\": \"The subject of the  
email\"\\n },\\n \"message\": {\\n \"type\":  
\"string\",\\n \"description\": \"The body of the  
email message\"\\n }\\n },\\n \"required\": [\\n  
\"recipient\",\\n \"subject\",\\n \"message\"\\n ]\\n  
},  
  {  
    "content": "I need to send an email to my boss. Can  
you help me with that?",  
    "role": "user"  
  },  
  {  
    "content": "Of course, I can help you with that.  
Could you please provide me with the recipient's  
email address, the subject of the email, and the  
message content?",  
    "role": "system"  
  }]
```

tools  
string · lengths

476+948 40.9%

null

text  
string · lengths

392+2.01k 65.7%

```
<start_of_turn>user You are a helpful assistant, with no  
access to external functions.<end_of_turn>..
```

```
[  
{  
  "type": "function",  
  "function": {  
    "name": "send_email",  
    "description": "Send an email to a recipient",  
    "parameters": {  
      "type": "object",  
      "properties": {  
        "recipient": {  
          "type": "string",  
          "description": "The email address of the recipient"  
        },  
        "subject": {  
          "type": "string",  
          "description": "The subject of the email"  
        },  
        "message": {  
          "type": "string",  
          "description": "The body of the email message"  
        }  
      },  
      "required": [  
        "recipient",  
        "subject",  
        "message"  
      ]  
    }  
  }]
```

```
<start_of_turn>user  
You are a helpful assistant with access to the following  
functions. Use them if required -  
[  
{  
  "type": "function",  
  "function": {  
    "name": "send_email",  
    "description": "Send an email to a recipient",  
    "parameters": {  
      "type": "object",  
      "properties": {  
        "recipient": {  
          "type": "string",  
          "description": "The email address of the recipient"  
        },  
        "subject": {  
          "type": "string",  
          "description": "The subject of the email"  
        },  
        "message": {  
          "type": "string",  
          "description": "The body of the email message"  
        }  
      },  
      "required": [  
        "recipient",  
        "subject",  
        "message"  
      ]  
    }  
  }]
```

# 자연어 처리의 기본

Hugging Face 모델 예시 소개

Hugging Face

Search models, datasets, users...

Models

Datasets

Spaces

Posts

Docs

Solutions

Pricing



## Spaces

Discover amazing AI apps made by the community!

Create new Space

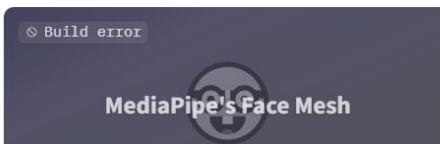
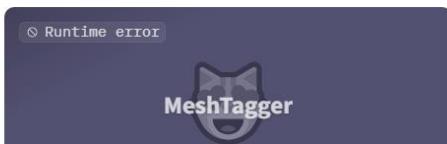
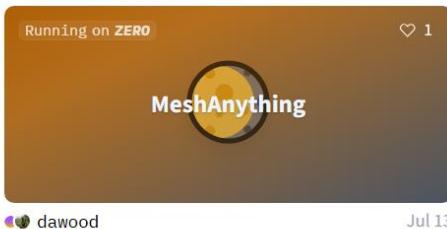
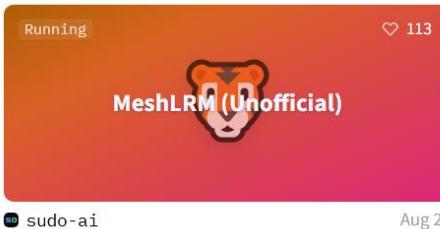
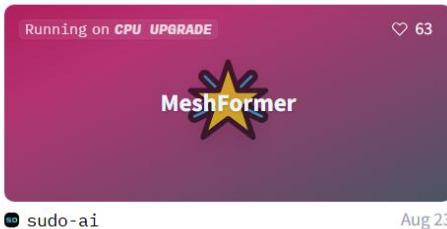
or [Learn more about Spaces](#)

mesh

Browse [ZeroGPU Spaces](#)

Full-text search

↑ Sort: Trending



[https://huggingface.co/new\\_space](#)

### Create a new Space

Spaces are Git repositories that host application code for Machine Learning demos. You can build Spaces with Python libraries like Streamlit or Gradio, or using Docker images.

Owner: mac999 / Space name: New Space name

Short description: Short Description

License: License

Select the Space SDK: You can choose between Streamlit, Gradio and Static for your Space. Or pick Docker to host any other app.

Streamlit (NEW)

Gradio (3 templates)

Docker (15 templates)

Static (3 templates)

# 자연어 처리의 기본

Hugging Face 모델 예시 소개

Spaces

shariqfarooq/ZoeDepth

like 685

Running on A10G

⋮

Depth Prediction

Image to 3D

360 Panorama to 3D

## Image to 3D mesh

Convert a single 2D image to a 3D mesh



Keep occlusion edges

Submit

# 자연어 처리의 기본 Hugging Face 모델 예시 소개

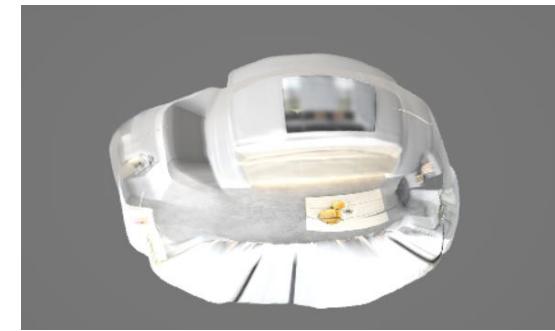
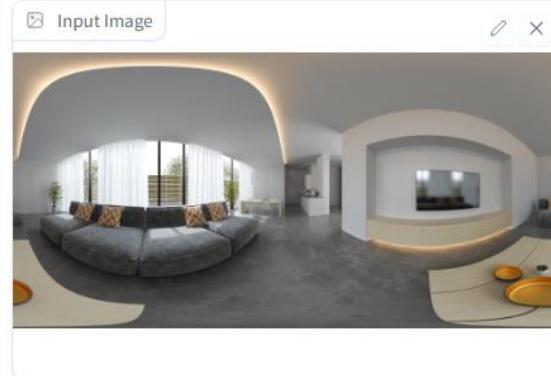


Depth Prediction   Image to 3D   360 Panorama to 3D

## Panorama to 3D mesh

Convert a 360 spherical panorama to a 3D mesh

ZoeDepth was not trained on panoramic images. It doesn't know anything about panoramas or spherical projection. Here, we just treat the estimated depth as radius and some projection errors are expected. Nonetheless, ZoeDepth still works surprisingly well on 360 reconstruction.



# Transformer와 최신 AI 모델

Transformer 구조 이해  
BERT, GPT 등의 동작 원리  
왜 Transformer가 표준이 되었는가?

# 최신 AI 모델

## Transformer 구조 이해 (Self-Attention, Positional Encoding)

### 트랜스포머(Transformers)

텍스트 생성 영역에서 GPT(Generative Pre-trained Transformer)와 같은 변환기 기반 모델은 방대한 양의 텍스트 데이터에서 어텐션(Attention)계산을 통해 패턴을 학습. 일관되고 맥락적으로 관련성 있는 토큰 시퀀스 학습 가능.  
조건화된 이미지 생성에 사용.

Ashish Vaswani\*  
Google Brain  
avaswani@google.com

Noam Shazeer\*  
Google Brain  
noam@google.com

Niki Parmar\*  
Google Research  
nikip@google.com

Jakob Uszkoreit\*  
Google Research  
usz@google.com

Llion Jones\*  
Google Research  
llion@google.com

Aidan N. Gomez\*  
University of Toronto  
aidan@cs.toronto.edu

Lukasz Kaiser\*  
Google Brain  
lukasz.kaiser@google.com

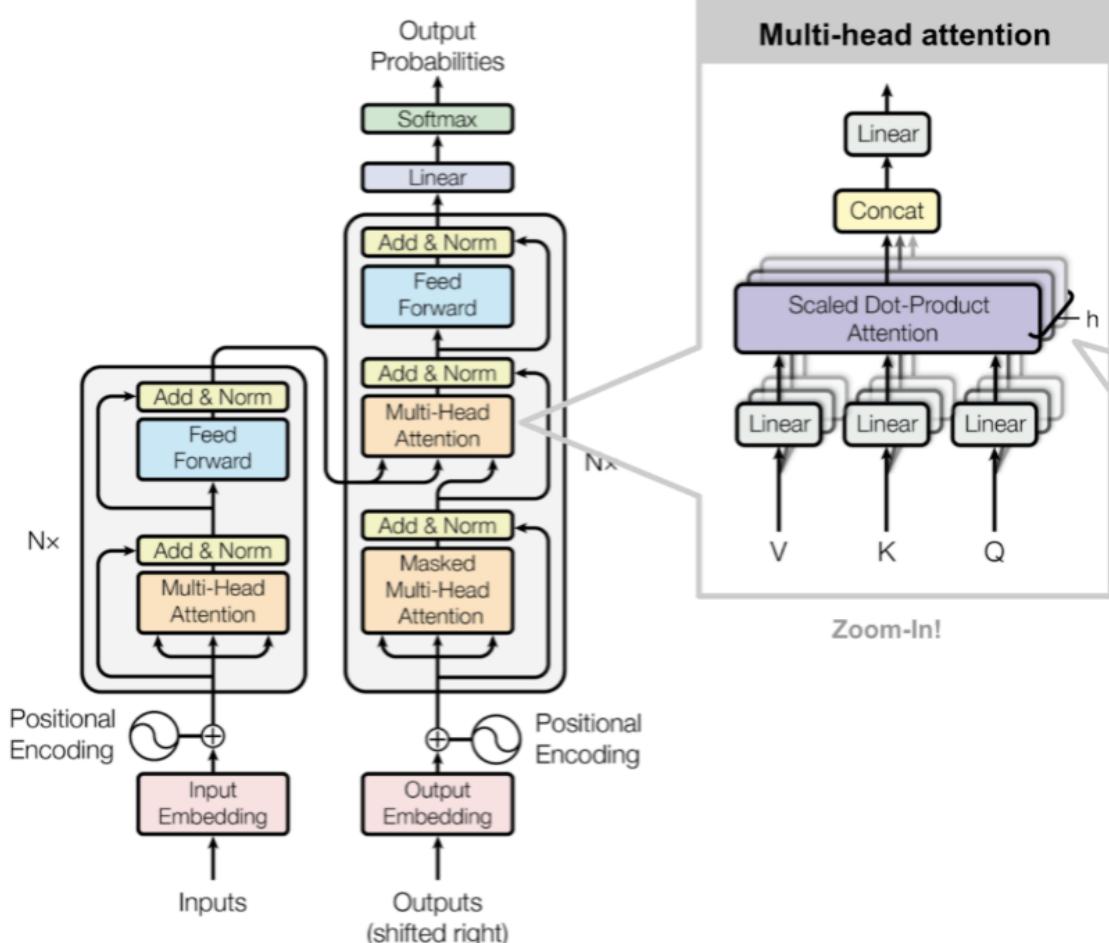
Illia Polosukhin\*  
illia.polosukhin@gmail.com

#### Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.

#### 1 Introduction

Recurrent neural networks, long short-term memory [12] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [29, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [31, 21, 13].



# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

## 어텐션(Attention)

시퀀스 내의 각 단어는 다른 모든 요소들과의 관계를 계산하여, 어떤 단어가 현재 처리 중인 단어 예측에 가장 큰 영향을 미치는지 가중치를 계산한 것

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

The FBI is chasing a criminal on the run .

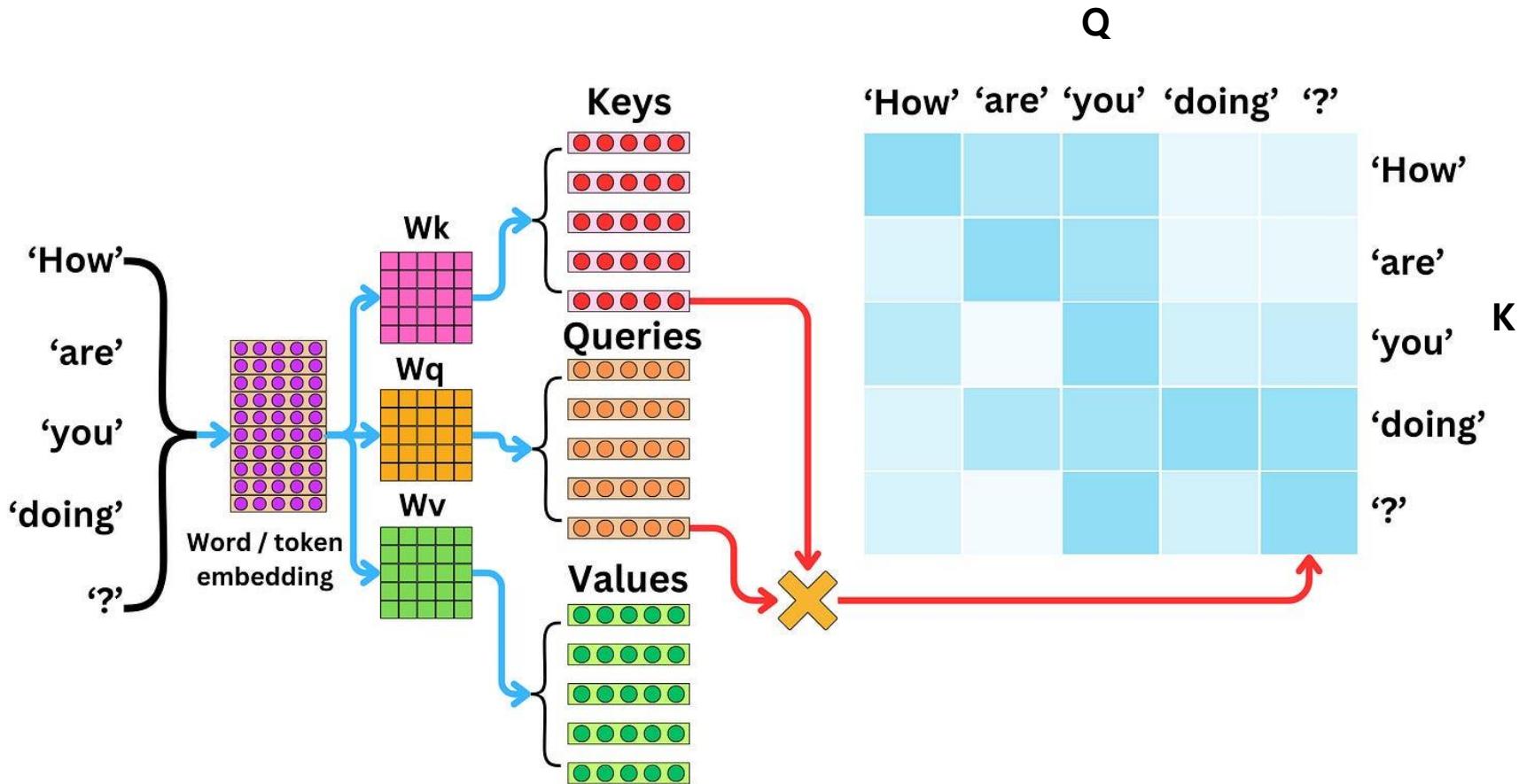
The FBI is chasing a criminal on the run .

주어진 문장에서 어텐션 예시(적색 표시는 현재 단어 토큰, 청색은 주의 집중되어 활성화되는 토큰을 표현. Long Short-Term Memory-Networks for Machine Reading, 2016)

# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

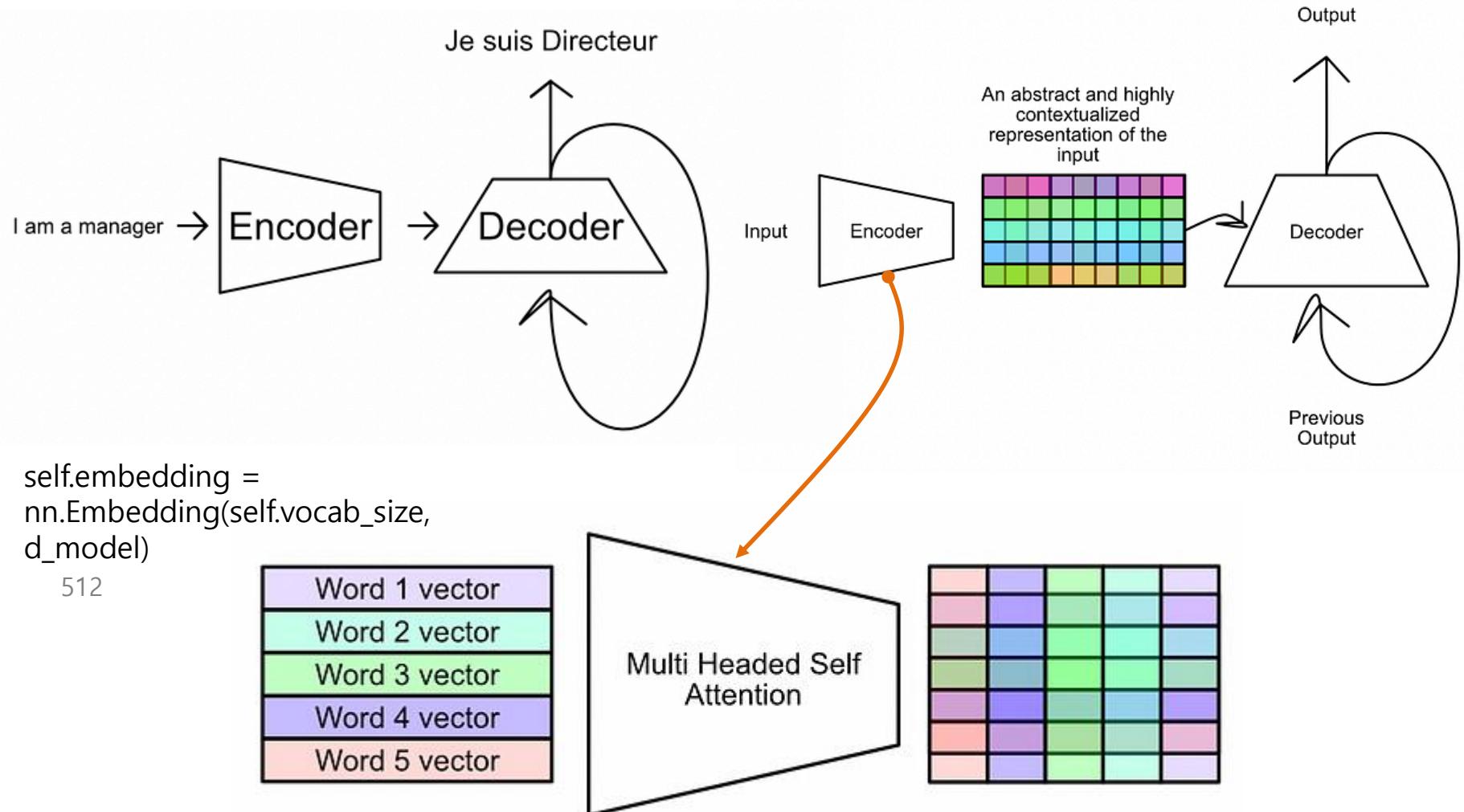
## 어텐션(Attention)



# 최신 AI 모델

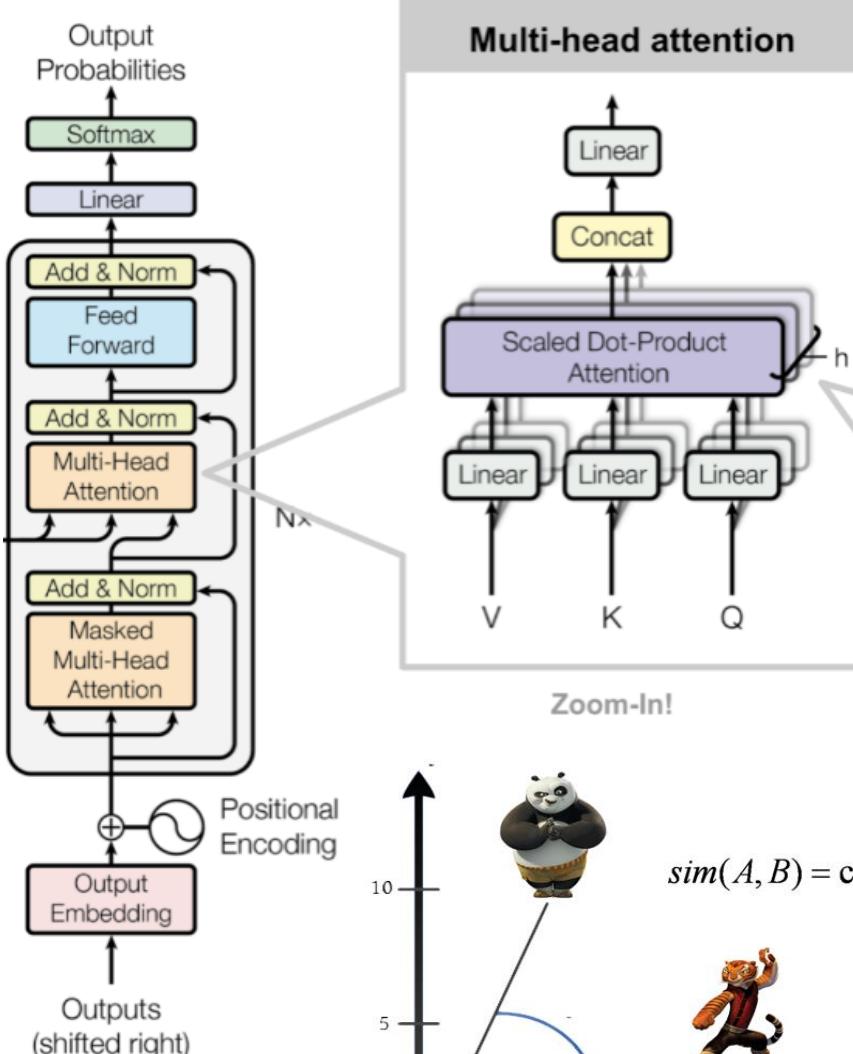
## Transformer 구조 이해 (Self-Attention, Positional Encoding)

### 트랜스포머(Transformers)

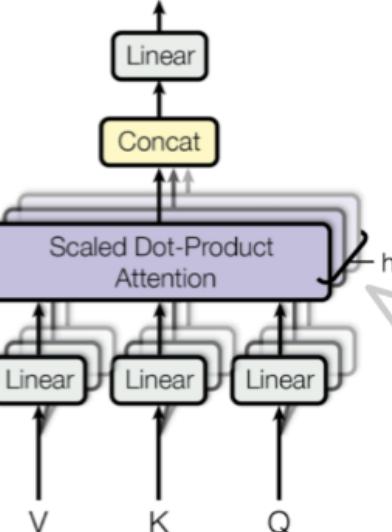


# 최신 AI 모델

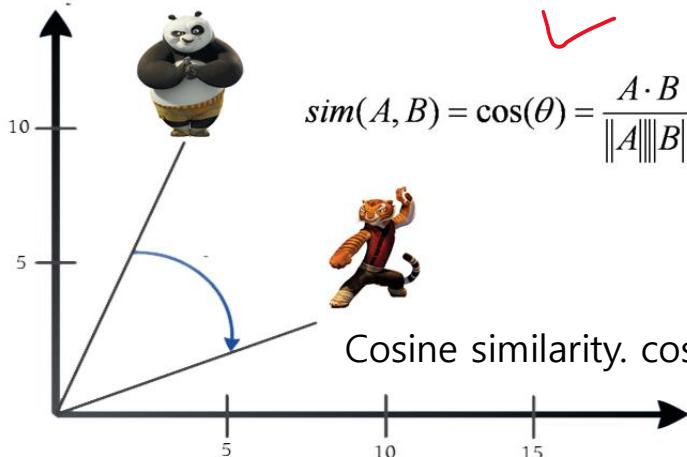
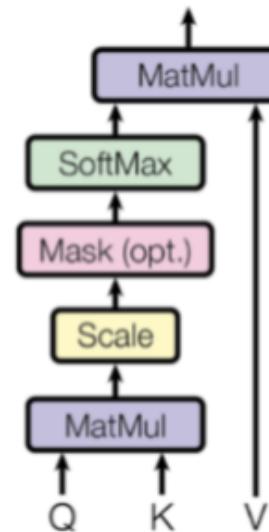
Transformer 구조 이해 (Self-Attention, Positional Encoding)



## Multi-head attention



Zoom-In!



Cosine similarity.  $\cos(0^\circ)=1$

ex) source code

$$V \times \text{Softmax}(QK^T)$$

Diagram illustrating the calculation of the attention matrix  $Z$  from  $Q$  and  $K^T$ . The  $Q$  matrix (purple) is multiplied by the transpose of the  $K$  matrix (orange) and then scaled by  $\sqrt{d_k}$  before being passed through a **Softmax** layer to produce the weight matrix  $Z$  (pink).

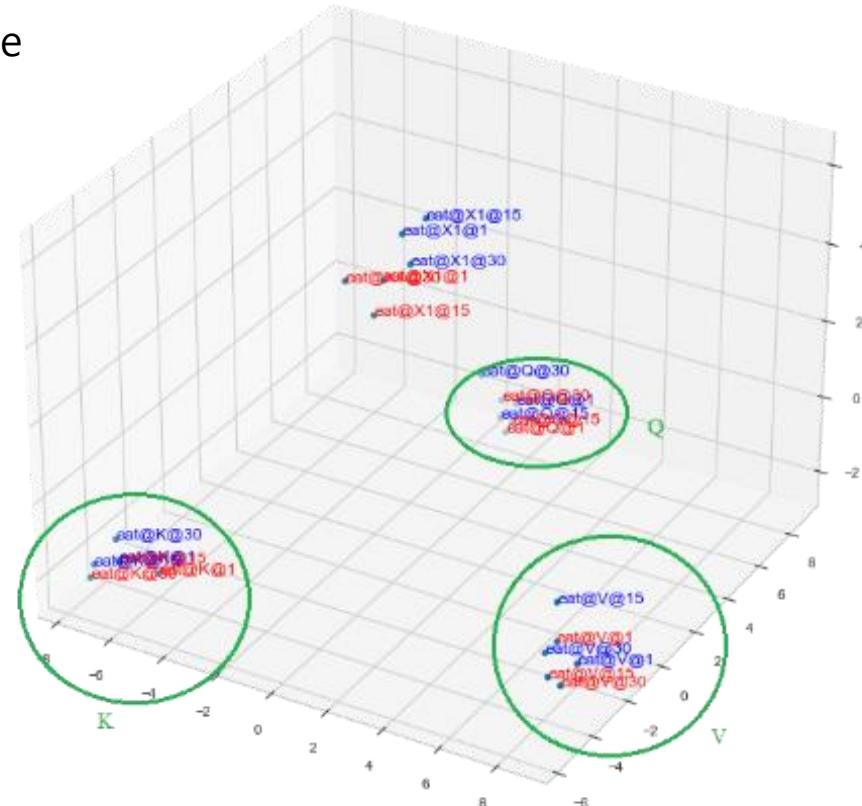
# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

- 처음엔  $QK^T$ 가 의미 없는 유사도를 계산함  $\rightarrow$  softmax 후  $V$ 를 평균해서 출력
- 이 결과가 예측 라벨(예: 다음 단어)과 멀면 loss 증가
- 역전파로  $Q, K, V$ 를 만드는 가중치  $W_Q, W_K, W_V$ 가 업데이트됨
- 이 과정이 수천만 문장을 반복하면서 각 QKV가 문맥에서 다른 역할을 하도록 학습됨

역할 = Query    Key    Value

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V$$



# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

## Contextual embedding

Self-Attention 을 통과한 벡터값은 어떤 문맥(문장)에 놓이느냐에 따라 그 의미를 반영하는 고유한 벡터표현을 가지게 됨. 예를 들어, "과일 사과"의 임베딩과 "행위 사과"의 임베딩은 주변 단어의 영향을 받아 서로 다른 벡터 값을 가지게 됨

"나는 학생" → 어텐션 → out → FFN → logits (10000차원) → softmax → "입니다" (예측)

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V$$

out

문맥 반영된 벡터 (차원: [seq\_len, d\_model])

### Feed-Forward Network (FFN)

각 out 벡터를 비선형적으로 변환 (정보 강화)

### 디코더 스택 마지막 출력

최종 출력 hidden state

### Linear + Softmax

각 hidden state에 대해 전체 vocabulary 크기만큼의 로짓(logits)을 만듦 → softmax를 적용해 확률로 변환

### 토큰 예측

확률이 가장 높은 단어가 다음 단어로 예측됨 (예: "student" 다음에 "is" 예측 등)

ex) source code

# 최신 AI 모델

## Transformer 구조 이해 (Self-Attention, Positional Encoding)

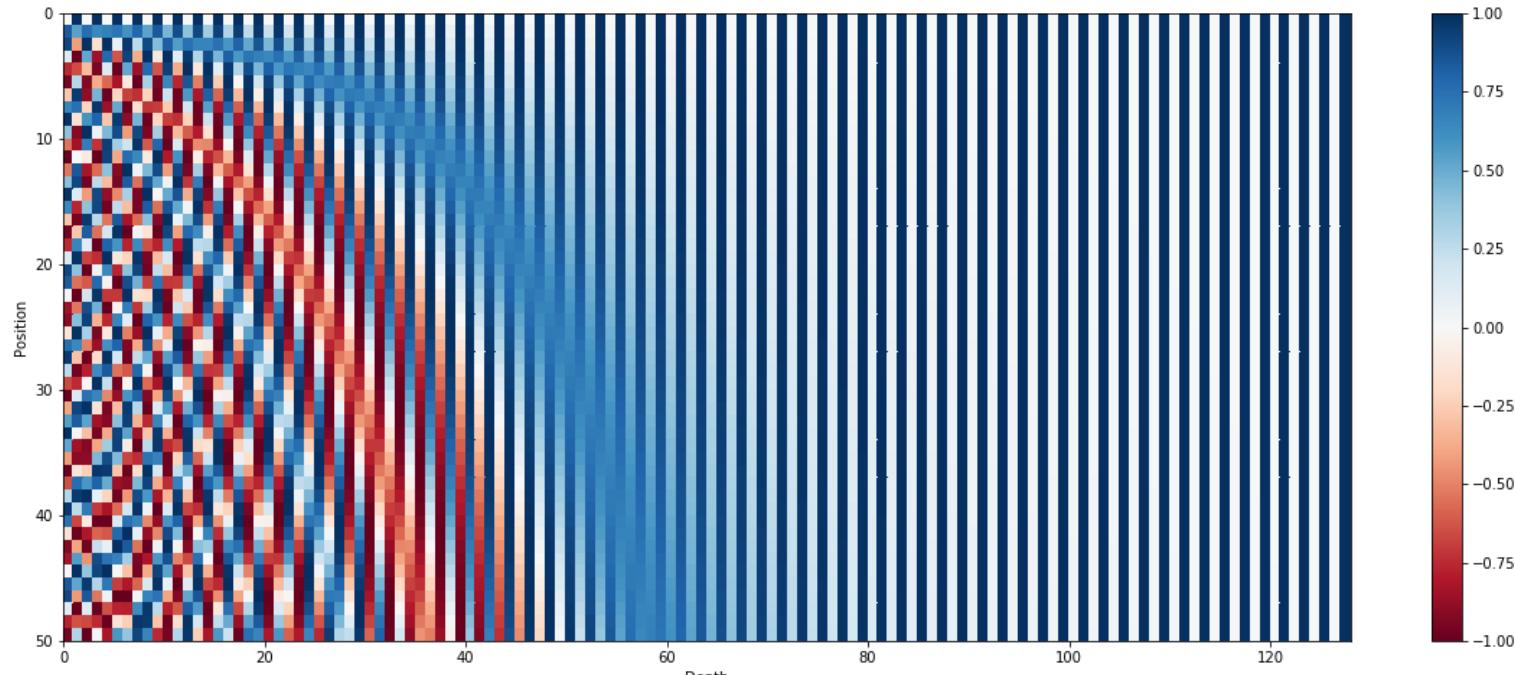
### Positional Encoding

트랜스포머는 순서를 고려하지 않기 때문에 각 단어의 위치 정보를 인코딩  
이를 위해 사인(sin), 코사인(cos) 함수를 기반으로 위치 인코딩 벡터를 생성

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$

여기서,  $i$  = 차원 인덱스  
 $d_{\text{model}}$  = 임베딩 길이  
 $pos$  = 시퀀스 내에서 단어의 위치



# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

## Scaled Dot Product Attention

이 모듈은 어텐션의 기본 동작을 수행하는 핵심 요소.  
입력으로 주어진 Q(Query), K(Key), V(Value) 벡터를 이용하여 다음 연산을 수행

## Multi-Head Attention

어텐션을 단일 벡터로 계산하면 정보 손실이 크기 때문에, 여러 개의 어텐션 "헤드"를 병렬로 실행한 후 결과를 concat하여 사용

## Positional Encoding

트랜스포머는 순서를 고려하지 않기 때문에 각 단어의 위치 정보를 인코딩  
이를 위해 사인(sin), 코사인(cos) 함수를 기반으로 위치 인코딩 벡터를 생성

```
class ScaledDotProductAttention(nn.Module):
    def forward(self, Q, K, V):
        scores = torch.matmul(Q, K.transpose(-2, -1)) / math.sqrt(d_k)
        attn = F.softmax(scores, dim=-1)
        return torch.matmul(attn, V)

class MultiHeadAttention(nn.Module):
    def forward(self, Q, K, V):
        # Q, K, V 선형 변환 및 split
        # 각 헤드별 attention 계산
        # concat 후 출력 선형 변환
        return output

class PositionalEncoding(nn.Module):
    def forward(self, x):
        # 위치별 sin, cos 벡터 계산 후 입력에 더함
        return x + self.pe[:, :x.size(1)]

class EncoderLayer(nn.Module):
    def forward(self, x):
        x = x + self_attn(x, x, x) # Residual
        x = LayerNorm(x)
        x = x + FFN(x) # Residual
        x = LayerNorm(x)
        return x
```

# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

## # 1. 입력 및 라벨 시퀀스

```
input_sentence = ["I", "am", "a", "student"]    # 영어 입력  
target_sentence = ["<sos>", "저는", "학생", "입니다"] # 한국어 출력 입력 (디코더 입력)  
target_labels = ["저는", "학생", "입니다", "<eos>"]   # 예측할 실제 정답 (디코더 출력)
```

## # 2. 인코더

```
X = embed(input_sentence)                      # 입력 시퀀스 임베딩 → [x_I, x_am, x_a,  
x_student]
```

```
Q_enc = linear_Q(X)                          # Q from encoder input  
K_enc = linear_K(X)                          # K from encoder input  
V_enc = linear_V(X)                          # V from encoder input
```

```
encoder_outputs = self_attention(Q_enc, K_enc, V_enc) # 인코더 출력: 입력 문맥이 반영  
된 벡터들
```

# 최신 AI 모델

Transformer 구조 이해 (Self-Attention, Positional Encoding)

## # 3. 디코더

```
Y = ["<sos>"]                      # 디코더 초기 입력 (<sos> 부터 시작)
logits_sequence = []                  # 각 시점의 출력 로짓을 저장

for t in range(len(target_labels)):

    Y_embed = embed(Y)              # 현재까지 생성된 디코더 입력 임베딩

    # 디코더 Self-Attention (마스킹 포함)
    Q_dec = linear_Q_dec(Y_embed)   # Q from decoder input
    K_dec = linear_K_dec(Y_embed)   # K from decoder input
    V_dec = linear_V_dec(Y_embed)   # V from decoder input
    dec_out = masked_self_attention(Q_dec, K_dec, V_dec)      # 디코더 내부 문맥 계산 (미래
                                                                # 마스킹 포함)

    # Cross-Attention: 디코더 → 인코더 입력 참조
    Q_cross = linear_Q_cross(dec_out) # Q from 디코더 문맥
    K_cross = linear_K_cross(encoder_outputs) # K from 인코더 출력
    V_cross = linear_V_cross(encoder_outputs) # V from 인코더 출력
    cross_out = attention(Q_cross, K_cross, V_cross) # 입력 문장 정보에 기반한 번역 벡터
```

# 최신 AI 모델

## Transformer 구조 이해 (Self-Attention, Positional Encoding)

```
for t in range(len(target_labels)):
```

```
...
```

```
# 출력 로짓 생성
```

```
logits = linear_output(cross_out)
```

```
logits_sequence.append(logits[-1])
```

```
# vocab 크기의 예측 로짓
```

```
# 마지막 위치 로짓만 저장 (가장 최근 토큰 예
```

```
측)
```

```
# 다음 디코더 입력을 위해 정답을 넣음 (teacher forcing)
```

```
next_token = target_sentence[t + 1]
```

```
Y.append(next_token)
```

```
# 예: "저는" 다음은 "학생"
```

```
# 다음 루프를 위한 디코더 입력에 추가
```

```
# 4. 손실(Loss) 계산
```

```
loss = 0
```

```
for logit, label_token in zip(logits_sequence, target_labels):
```

```
label_id = vocab_id(label_token)
```

```
# 정답 토큰을 ID로 변환
```

```
loss += cross_entropy_loss(logit, label_id)
```

```
# 예측 분포와 정답 ID 간의 CrossEntropy Loss
```

```
loss = loss / len(target_labels)
```

```
# 평균 손실값 (시퀀스 길이만큼 나눔)
```

```
# 5. 역전파 및 파라미터 업데이트 수행
```

# 최신 AI 모델

## Transformer 구조 이해 (Self-Attention, Positional Encoding)

instruction string · lengths	input string · lengths	output string · lengths
 <p>18                    379</p>	 <p>0                    1.42k</p>	 <p>10                    5.93k</p>
Find the most frequently used word in the given paragraph.	The cat is black, the dog is brown and the cow is white.	The: 3 times.
Generate a regex pattern to match the following string	String: "Hello123"	\w{5}[0-9]{3}
Write a code snippet to update the value of an element in an array.	Array: {3, 2, 5, 7, 8, 1} Index: 3 New value: 6	array[3] = 6;
Predict a label for the given sentence.	The phone is red.	Label: Color.
Print out the 5th element of an array.	arr = [55, 32, 10, 11, 8]	print(arr[4])
Classify the following text into two categories: 'Sports' and...	The critically acclaimed film 'Avatar' is about a paraplegic Marine who embark...	Entertainment
Create a formula in Excel to calculate the total cost given the price, tax,...	Price: 10 Tax: 0.08 Shipping: 6	=10+0.08*10+6
Classify the following given expression. Your response should be...	x = 0.5	float literal
Classify the following code as either a python function or code section.	x = "hello" z = x + " world"	Code section.
Generate a regex expression to match strings of length 8 with given...	Allowed letters: A-F and digits: 0-9	^[A-F0-9]{8}\\$

# 최신 AI 모델 Transformer 구조 이해 (Self-Attention, Positional Encoding)

Datasets: juneelee/sharegpt\_deepl\_ko like 53

Dataset card Viewer Files and versions Community

## Dataset Preview

API

View in Dataset Viewer

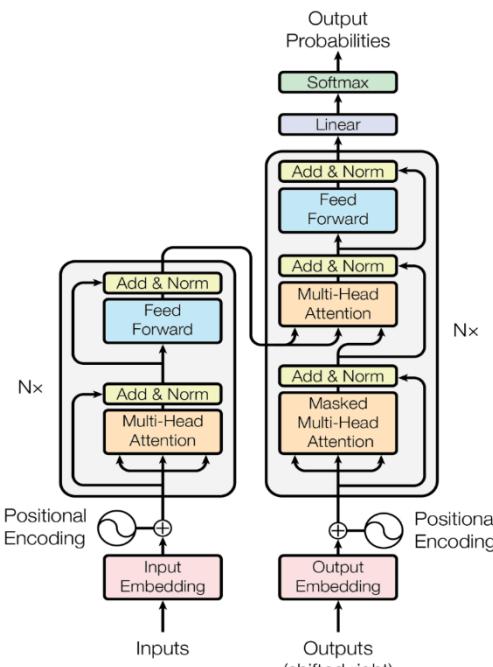
Split (1)

train

The full dataset viewer is not available (click to read why). Only showing a preview of the rows.

input string	instruction string	output string
	토니 로빈스가 설명하는 인간의 6가지 욕구를 요약해 보세요...	토니 로빈스는 인간의 행동과 동기를 이끄는 6가지 핵심 욕구를 설명합니다. 이 여섯 가지 욕구는 다음과 같습니다: 1. 확실성: 안전, 안정성, 예...
	고객 세그먼트가 잘 세분화되었는지 어떻게 알 수 있을까요? 세 가지로 요약할 수 있습니다.	1. 동질성: 세그먼트는 유사한 특성과 행동을 공유하는 고객으로 구성되어야 합니다. 2. 차별성: 세그먼트는 특성 및 행동 측면에서 다른 세그먼트...
	n Java에서 "이것은 {장소}에 있는 새로운 {객체}입니다"와 같은 문자열을 맵, {객체: "학생", "지점 3, 4"}로 바꾸고 "미 학생은 지점 3, 4...	을 사용하여 문자열의 자리 표시자를 맵의 값으로 바꿀 수 있습니다. 다음은 이를 수행하는 방법을 보여주는 코드 스ニ펫 예시입니다. ***java...
	전체 단락을 이와 같은 스타일로 작성하세요: 신들의 은총으로 은유적 언어의 신비롭고 수수께끼 같은 예술이 소환되어 우리 앞에 놓인 지침의 당혹...	보세요! 신성한 개입의 은총으로, 이해할 수 없고 불가사의한 은유적 언어의 예술이 우리 앞에 놓인 지침의 불가해한 전달 방식을 해명하기 위해 호...
	길이를 변경할 수 없습니다.	지식의 구도자 여러분, 잠시만요, 우리 앞에 제시된 지침의 복잡한 전달 방식을 설명하는데 사용된 은유적 언어의 난해하고 수수께끼 같은 예술에 ...
	계속	은유적 언어의 사용은 단순해 보이는 이 지침에도 신비감과 초월성을 불어넣어 줍니다. 특히 '회상하다'라는 단어는 이 지침에 숨겨진 지식이나 비밀...
	다음과 같은 C++ 함수가 있습니다: void add\_player(벡터& 플레이어)	주어진 C++ 함수의 'dummy' 변수는 'cin' 문을 사용하여 'dummy'라는 저스트 스트림을 이용해 이전에 나온 값에 모자르거나 미비한

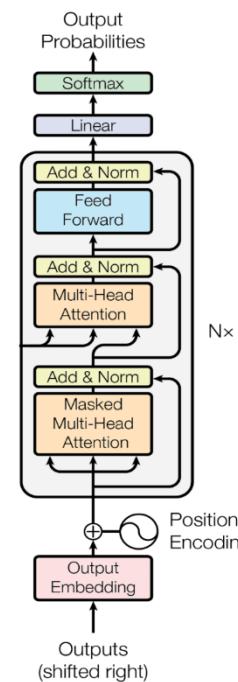
### Transformer



Encoder

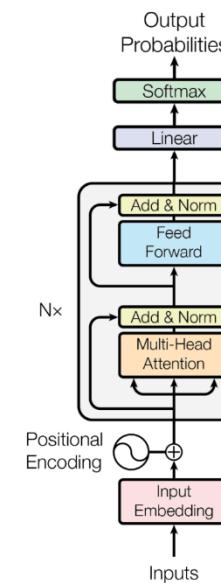
Decoder

### GPT\*



Decoder-only

### BERT\*



Encoder-only

# 최신 AI 모델

BERT, GPT 등의 동작 원리

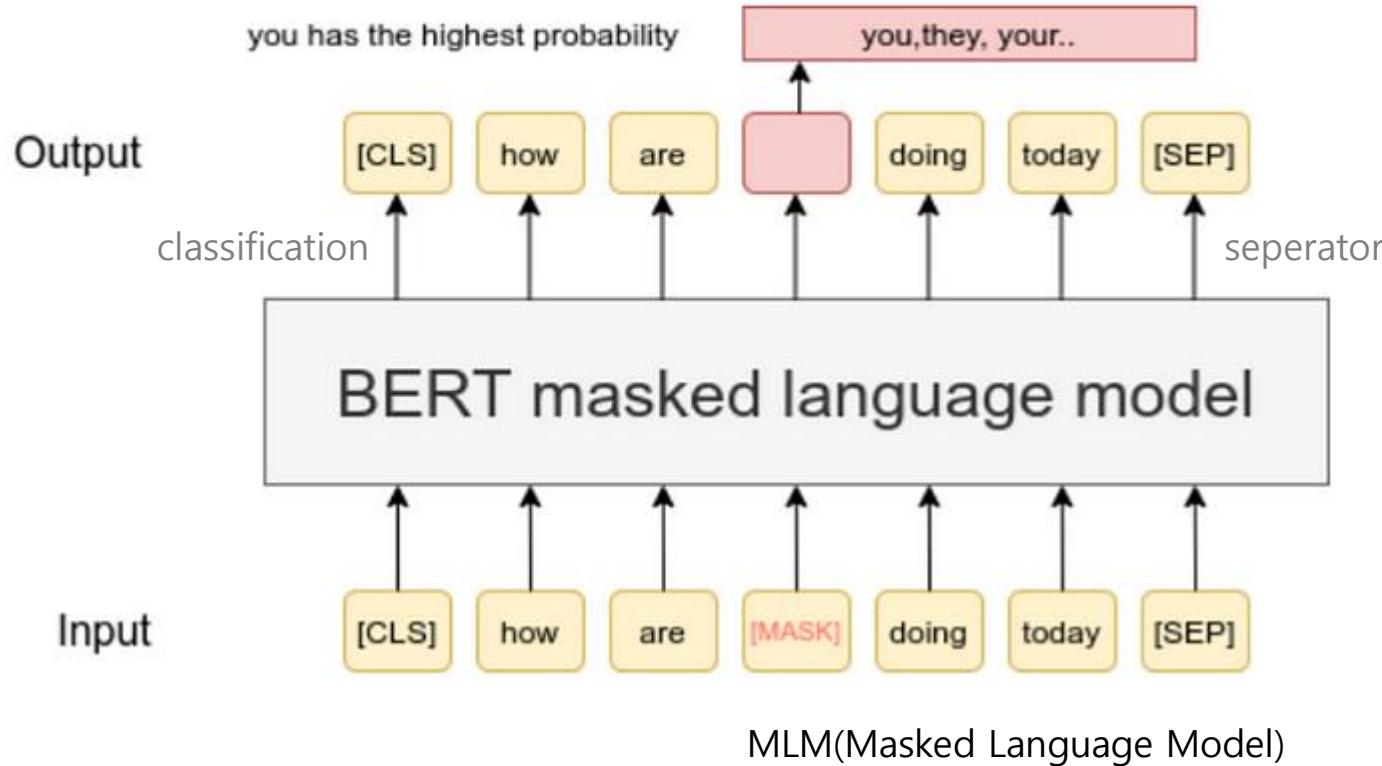


BERT

## BERT

BERT(Bidirectional Encoder Representations from Transformers)는 문장에서 단어들 간 문맥적 관계를 학습하는 기술. 이런 이유로, 트랜스포머의 인코더만 사용함. 자기 주의(Self attention)라는 메커니즘을 통해 문장 앞과 뒤 문맥을 기반으로, 각 단어의 중요성을 계산

BERT는 NER, 감정 분석 등과 같은 NLP 작업의 하위 집합에 가장 적합



# 최신 AI 모델

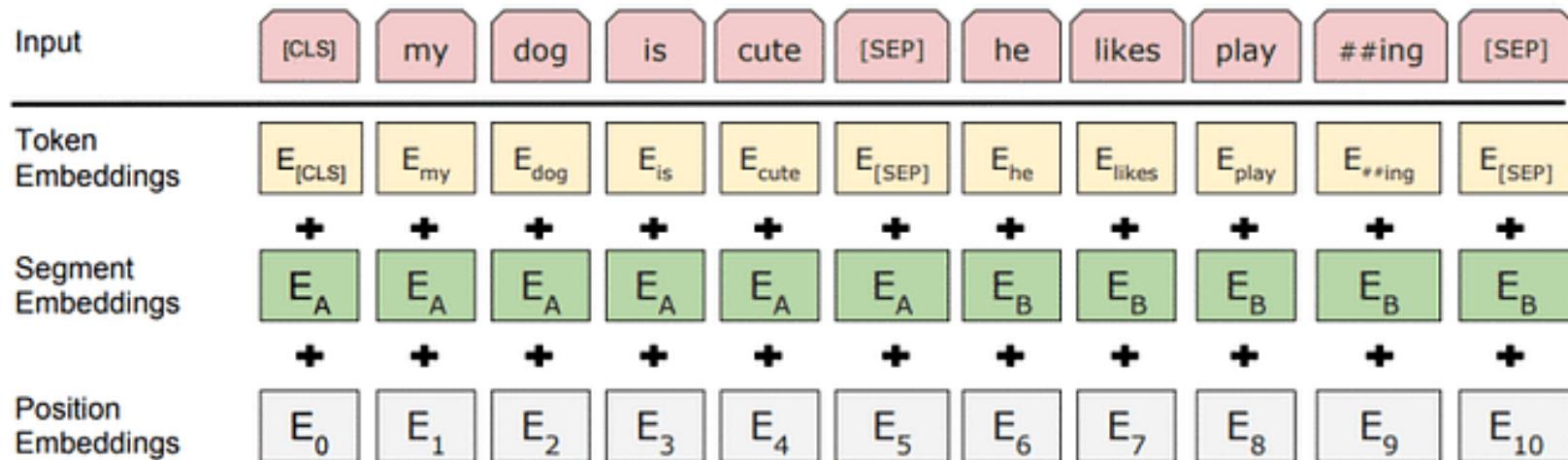
BERT, GPT 등의 동작 원리

## BERT

토큰 임베딩: [CLS] 토큰은 첫 번째 문장의 시작 부분에 있는 입력 단어 토큰에 추가. [SEP] 토큰은 각 문장의 끝에 삽입

세그먼트 임베딩: 문장 A 또는 문장 B를 나타내는 마커가 각 토큰에 추가. 이를 통해 인코더는 문장을 구별

위치 임베딩: 문장에서 해당 위치를 나타내기 위해 각 토큰에 위치 임베딩이 추가



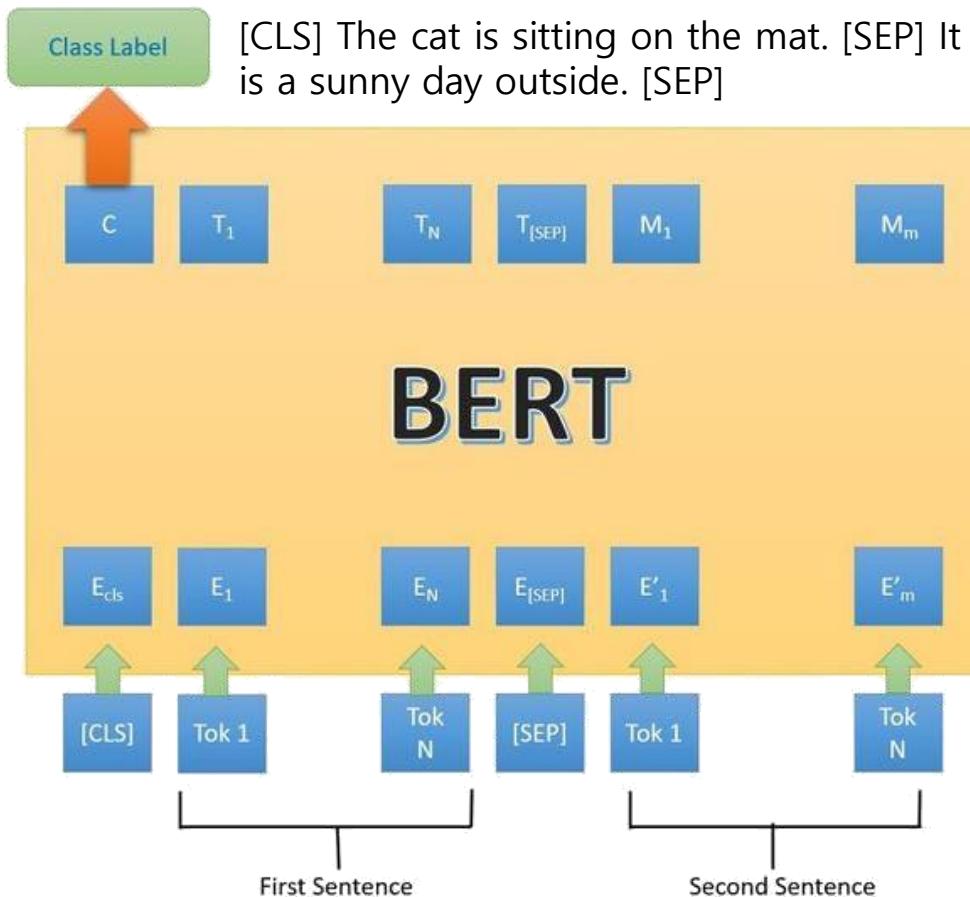
[google-research/bert: TensorFlow code and pre-trained models for BERT](https://google-research/bert)

# 최신 AI 모델

BERT, GPT 등의 동작 원리

## BERT

NSP(Next Sentence Prediction)를 위해 두 문장이 원본 텍스트에 연속적으로 나타나는지 여부를 예측하도록 훈련. 두 문장 사이에 특수 구분 기호 토큰([SEP])을 삽입. 또한 특수 분류 토큰([CLS])이 문장 시작 부분에 삽입되어 학습됨



[PAD]

입력 시퀀스 총 토큰 수를 최대 512개 까지 가져오는 데 사용되는 패딩 토큰

[UNK]

BERT의 어휘에 없는 토큰

[CLS]

분류 토큰

[SEP]

단일 입력 시퀀스에서 두 세그먼트를 구별하는 데 사용. 최대 2개까지 가능

[MASK]

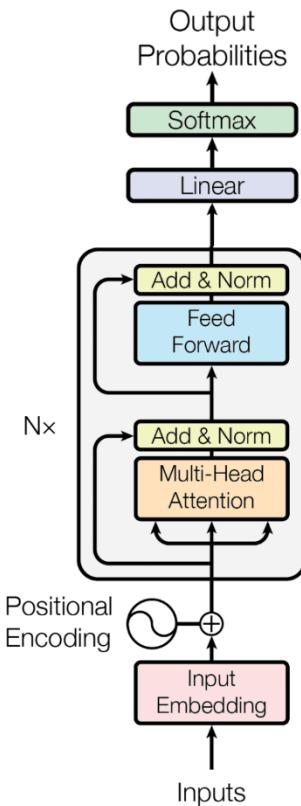
마스킹된 시퀀스에 대한 추론을 수행하는 데 사용되는 토큰

[What is BERT? How it is trained ? A High Level Overview | by Suraj Yadav | Medium](#)

# 최신 AI 모델

BERT, GPT 등의 동작 원리

## BERT



### BERT Base

### BERT Large

**Number of embedding dimensions,  $d_{model}$**

768                    1024

**Number of encoder blocks,  $N$ :**

12                    24

**Number of attention heads per encoder block:**

12                    16

**Size of hidden layer in feedforward network:**

3072                    4096

**Total parameters:**

110 million            340 million

# 최신 AI 모델

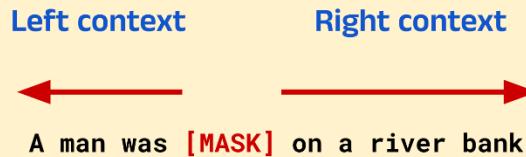
BERT, GPT 등의 동작 원리

## GPT

GPT는 OpenAI에서 개발. GPT는 새 단어를 한 번에 하나씩 예측하여, 출력 시퀀스를 생성하는 것이 목표. 예측된 각 단어는 후속 단어(오른쪽 문맥)가 아직 생성되지 않았기 때문에 이전 단어(왼쪽 문맥)가 제공하는 문맥만 활용함

### BIDIRECTIONAL CONTEXT

When predicting words within a sequence, all of the surrounding words can be used to gain contextual information.



### UNIDIRECTIONAL CONTEXT

When predicting future words, only the previous words can be used to gain contextual information.

Left context



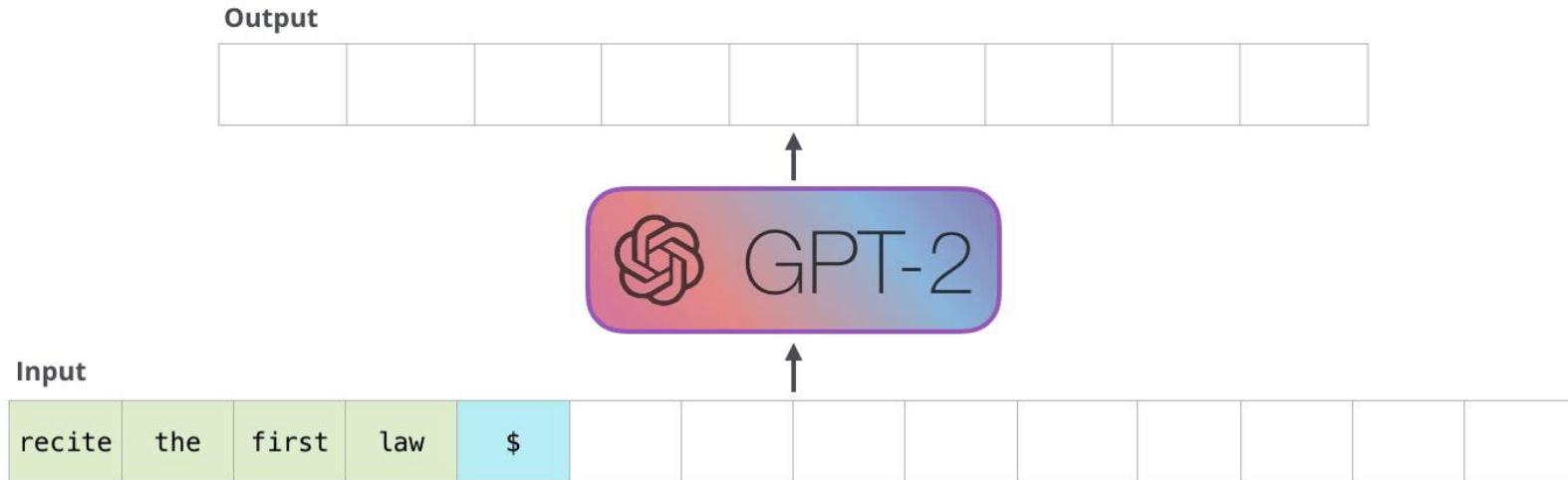
Write a poem about a man fishing on a river bank. Upon a [NEXT TOKEN]

# 최신 AI 모델

BERT, GPT 등의 동작 원리

## GPT

GPT는 자동 회귀 모델 방식으로 동작. 과거 값을 기반으로 미래 값을 예측. 각 토큰이 생성된 후 해당 토큰이 입력 시퀀스에 추가함



[Step by Step into GPT. GPT stands for Generative Pre-Training... | by Yan Xu | Medium](#)

# 최신 AI 모델

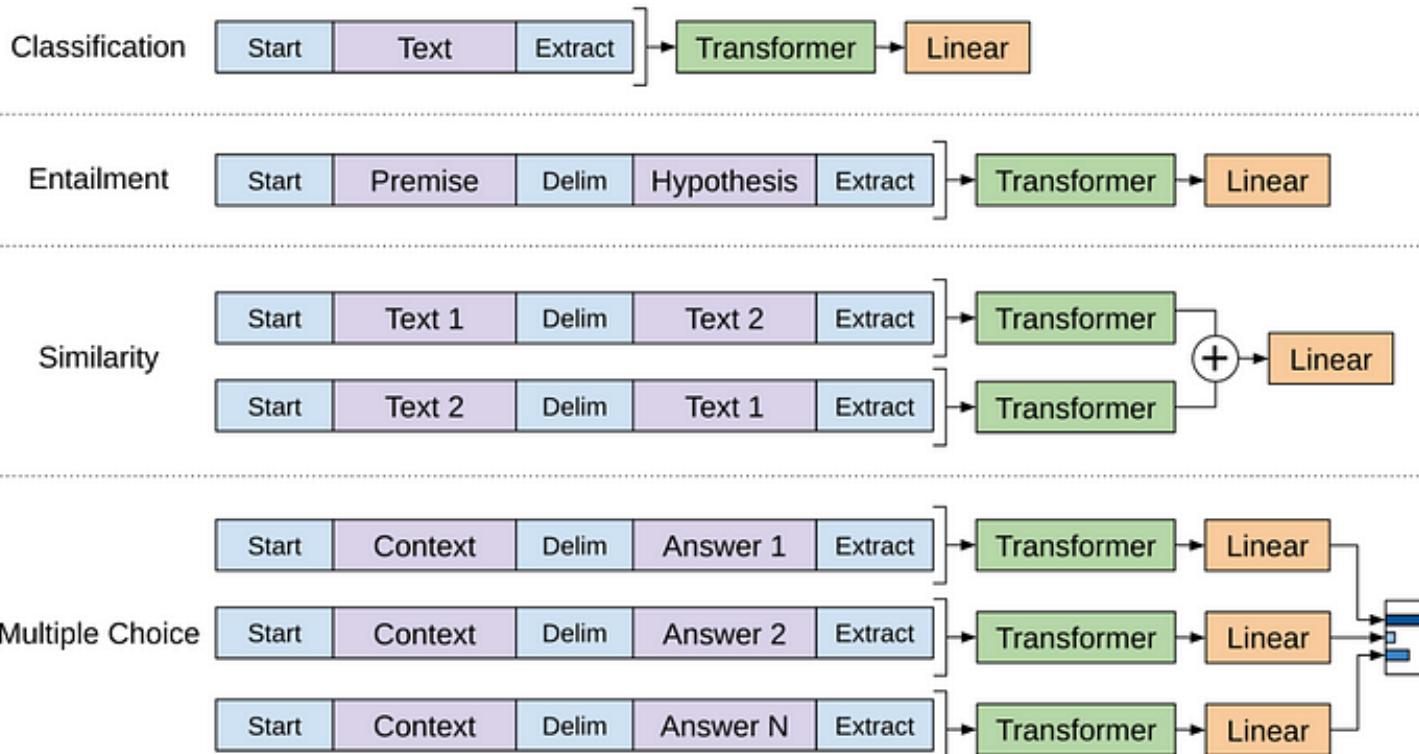
BERT, GPT 등의 동작 원리

## GPT

텍스트 분류 같은 작업의 경우 선형+소프트맥스 레이어를 추가하여 모델을 직접 미세 조정 가능  
질문 답변 또는 텍스트 포함과 같은 다른 작업에는 정렬된 문장 쌍 또는 문서, 질문 및 답변의 삼중항과 같은 구조화된 입력 가능

GPT는 순회 접근 방식 사용. 이런 방식 통해 작업 전반에 걸쳐 아키텍처를 크게 변경하지 않아도 됨  
모든 변환에는 초기화된 시작 토큰, 여러 문장 입력을 구분하는 구분 기호 및 끝 토큰을 포함

예를 들어, Start 및 Extract 토큰으로 둘러싸인 전제와 가설을 사용하여 하나의 시퀀스로 학습 가능. 여러 트랜스포머를 사용하고 객관식 작업을 위해 마지막 레이어의 출력을 병합할 수도 있음



Step by Step into  
GPT. GPT stands for  
Generative Pre-  
Training... | by Yan  
Xu | Medium

# 최신 AI 모델

BERT, GPT 등의 동작 원리

## GPT

GPT 구조는 큰 변화 없이 다양한 유스케이스 사용 가능함

### Training Dataset

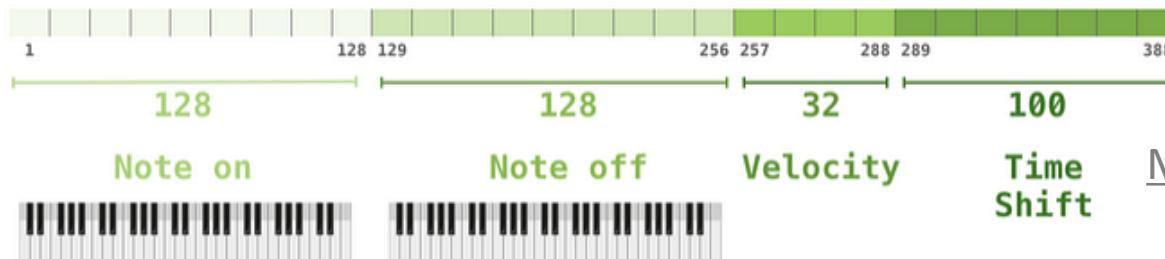
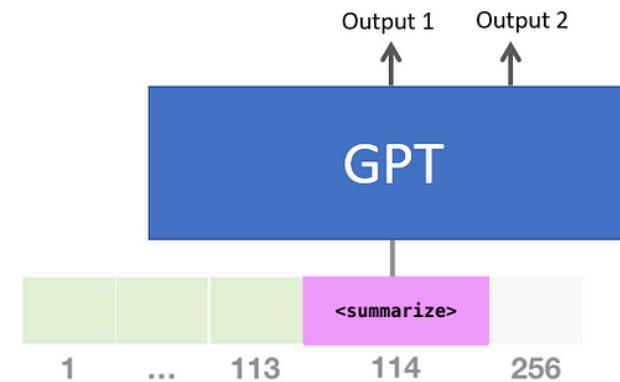
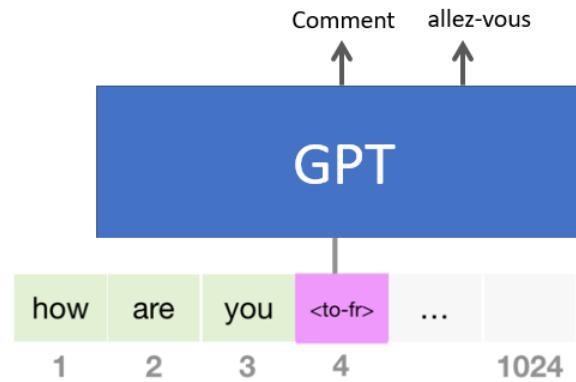
Eng to Fr

I	am	a	student	<to-fr>	je	suis	étudiant
let	them	eat	cake	<to-fr>	Qu'ils	mangent	de
good	morning	<to-fr>	Bonjour				

### Training Dataset

Summary

Article #1 tokens	<summarize>	Article #1 Summary
Article #2 tokens	<summarize>	Article #2 Summary
Article #3 tokens		<summarize>
		Article #3 Summary



[Step by Step into GPT](#). GPT stands for Generative Pre-Training... | by Yan Xu | Medium

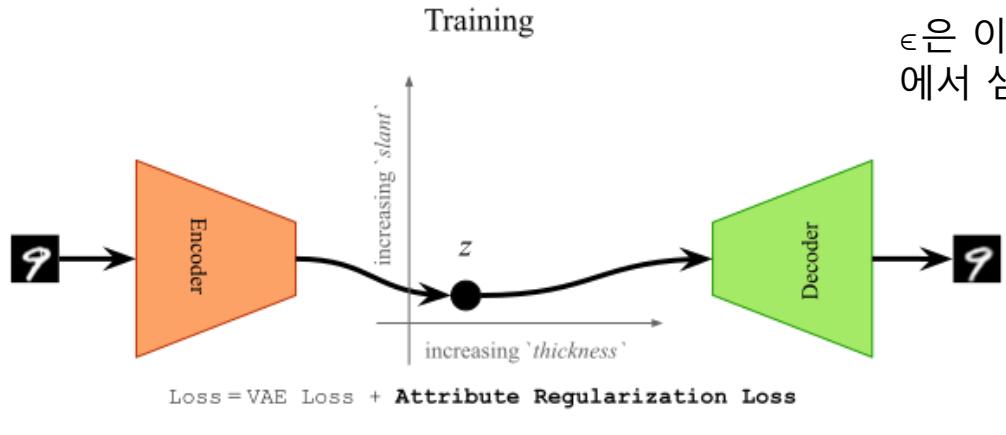
# 최신 AI 모델 주요 아키텍처 VAE

## 변이 자동 인코더(VAE. Variational autoencoder)

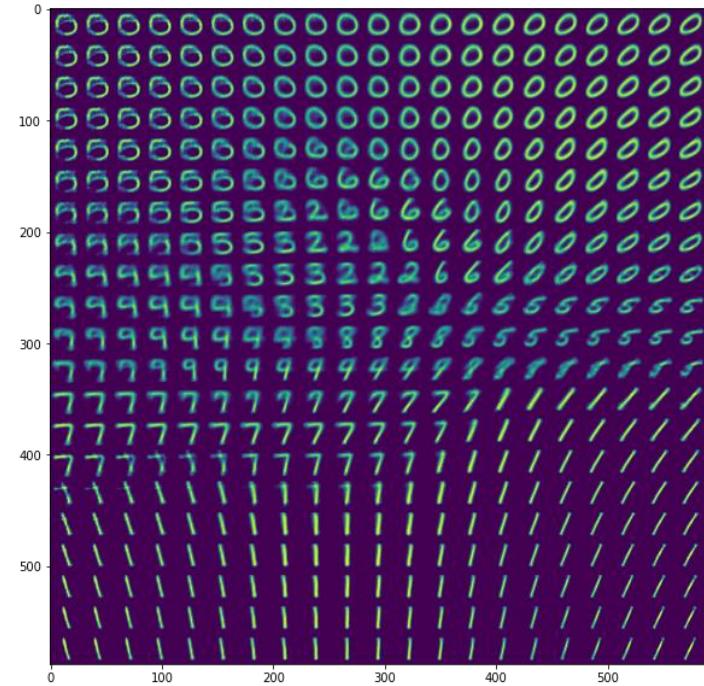
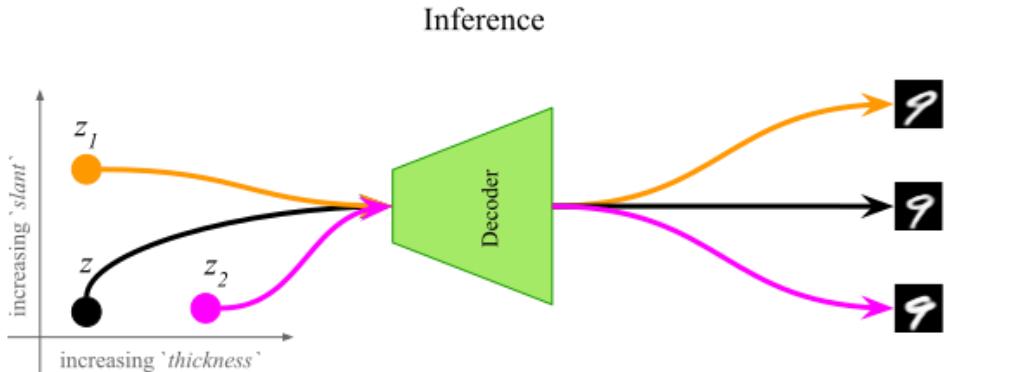
VAE는 압축된 잠재 공간에 데이터를 인코딩한 다음 다시 데이터로 디코딩하는 방법을 학습하는 생성 모델. 이는 특히 입력 데이터의 변형을 생성하는 데 유용

인코더는 입력 데이터  $X$ 를 받아서 단순히 하나의 잠재 벡터를 출력하는 대신, 잠재 공간의 확률 분포(평균  $\mu$ 와 분산  $\sigma^2$ )를 나타내는 두 개의 벡터를 출력. 디코더는 인코더가 출력한 평균과 분산으로부터 잠재 벡터  $z$ 를 직접 샘플링

$$\text{잠재공간 벡터 } z = \mu + \sigma \cdot \epsilon$$



$\epsilon$ 은 이 식이 미분 가능하도록 표준 정규 분포( $N(0,1)$ )에서 샘플링된 노이즈. 참고로,  $z = N(\mu, \sigma^2)$ 은 미분 불가



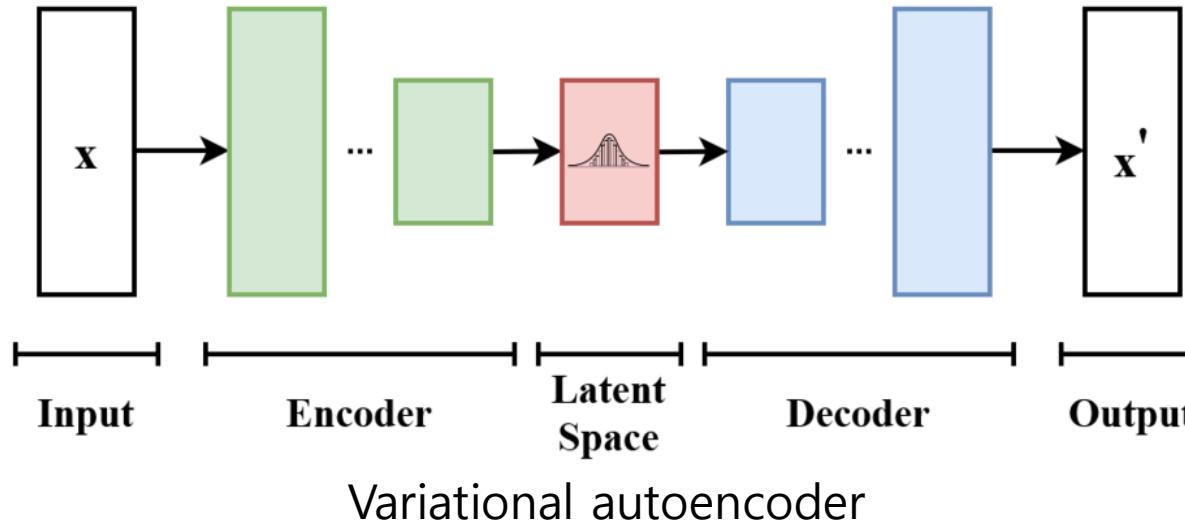
# 최신 AI 모델

주요 아키텍처 Latent Space

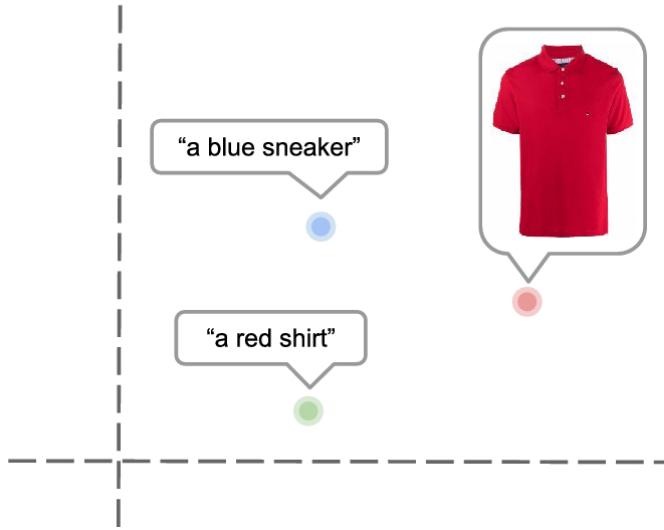
## 변이 자동 인코더(VAE. Variational autoencoder)



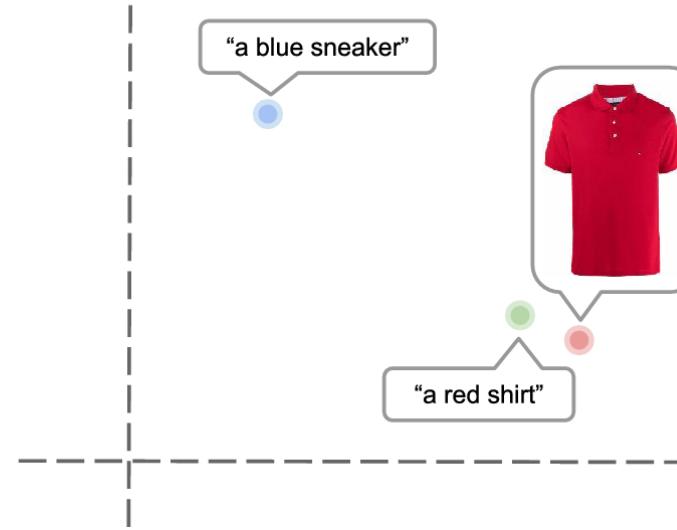
Villa savoye,  
1929, Le Corusier



Latent Space Before Training



Latent Space After Training

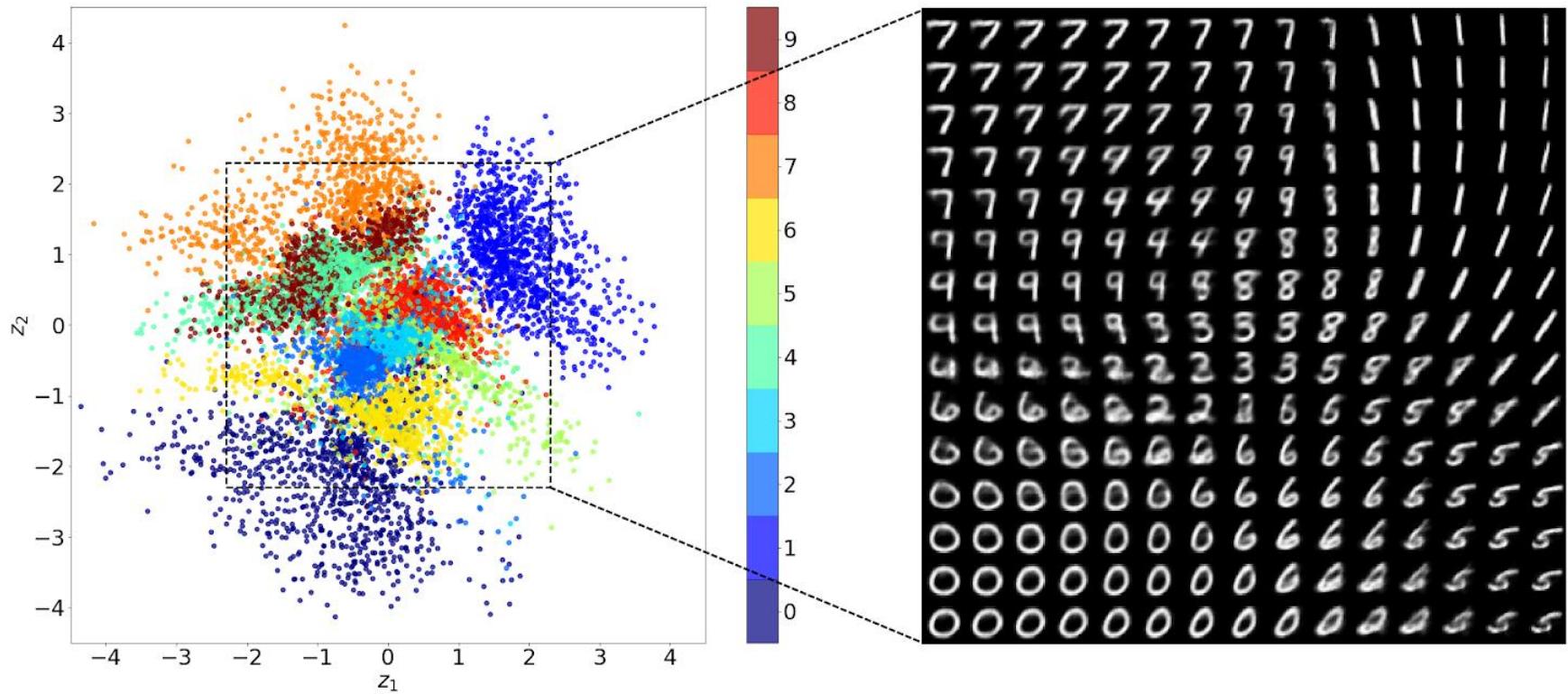


# 최신 AI 모델

## 주요 아키텍처 Latent Space

### 부드러운 다양체(smooth manifold) 효과

$(z = \mu + \sigma \cdot \epsilon)$ 은 이 잠재 공간에 노이즈를 주입하여, 인코더가 단순히 데이터를 특정 점에 맵핑하는 것을 넘어 '분포'를 학습



잠재공간에 맵핑된(인코딩된) 데이터(Alexej Klushyn, 2019.12, Learning Hierarchical Priors in VAEs)

# 최신 AI 모델

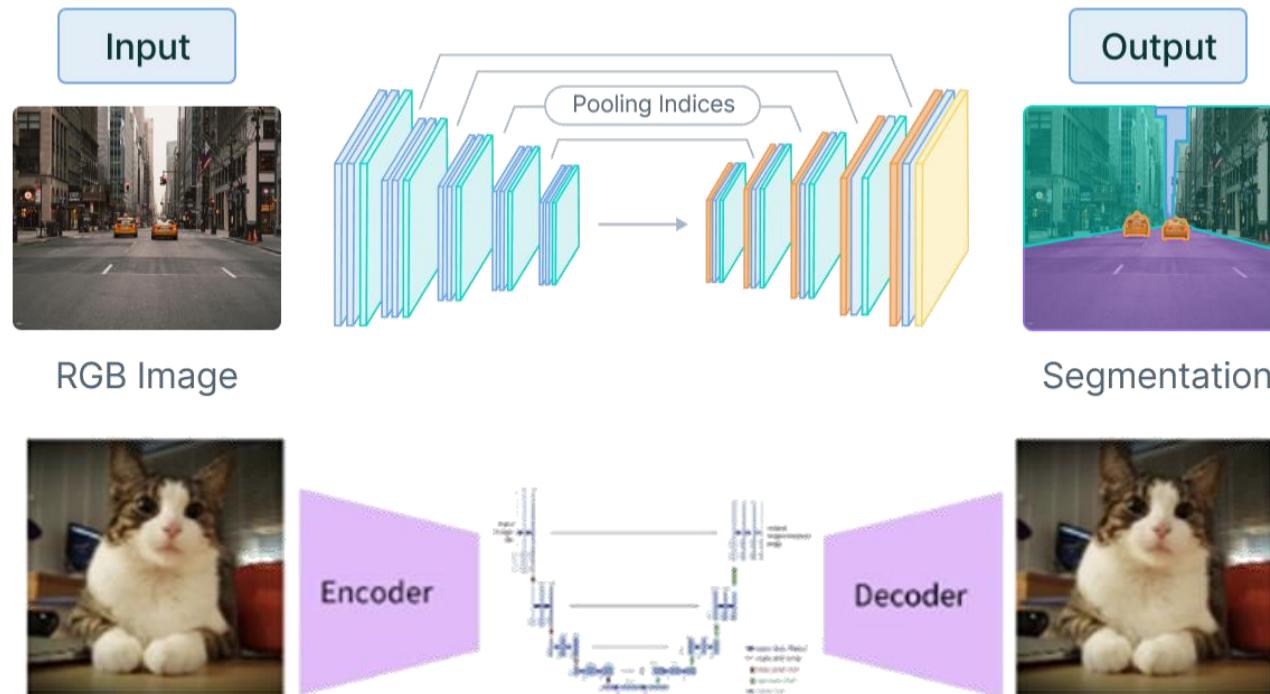
주요 아키텍처 U-Net

## U-Net

이미지를 압축하는 인코더와 이를 다시 확장하는 디코더 경로로 구성

인코더에서 디코더로 정보를 직접 전달하는 스킵 연결을 통해 이미지의 세부 정보를 보존, 픽셀 단위의 정확한 이미지를 생성함

## Convolutional encoder-decoder



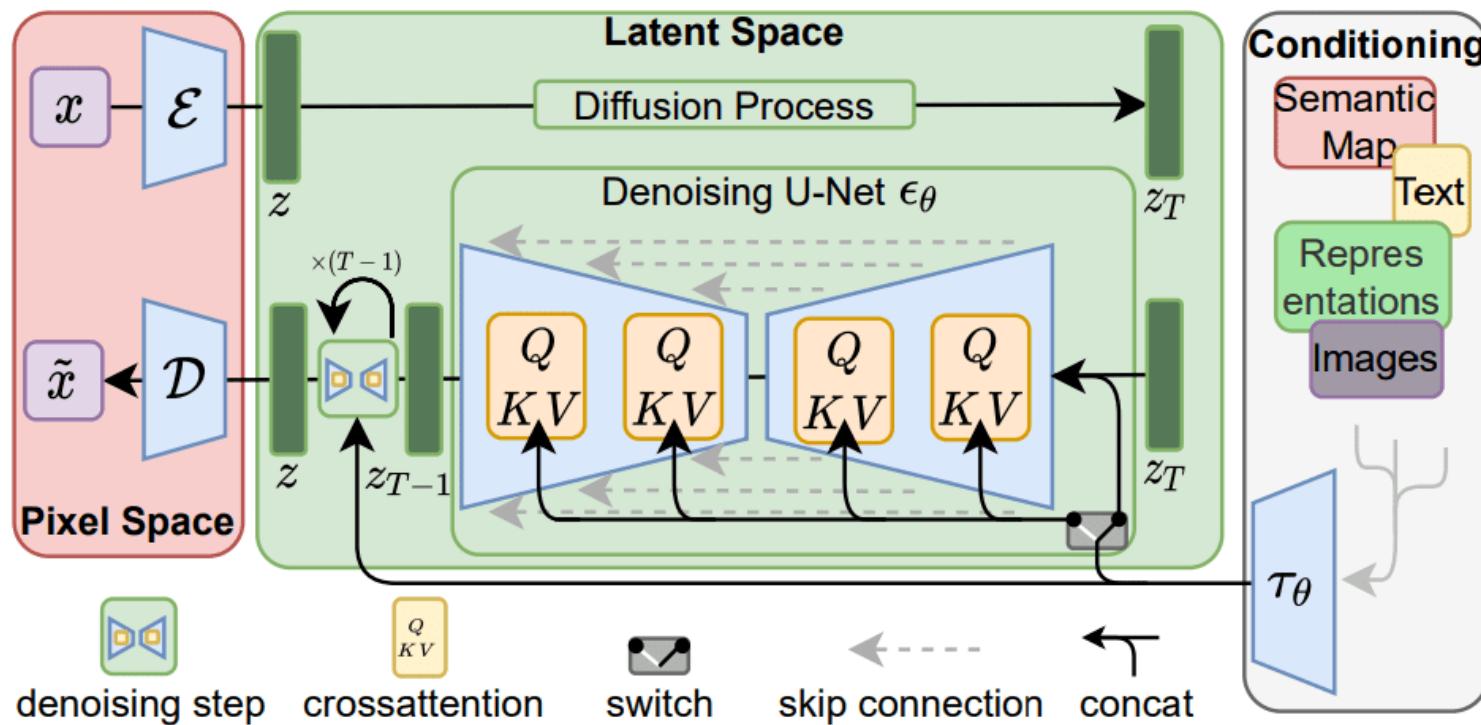
# 최신 AI 모델

왜 Transformer가 표준이 되었는가?

## Latent Diffusion Model

이미지를 텍스트 조건에 맞게 생성하는 잠재 확산 모델. 입력 이미지를 압축, 잠재 공간으로 변환한 후, 잠재공간의 노이즈가 섞인 표현  $z$  으로부터 점진적으로 노이즈를 제거함

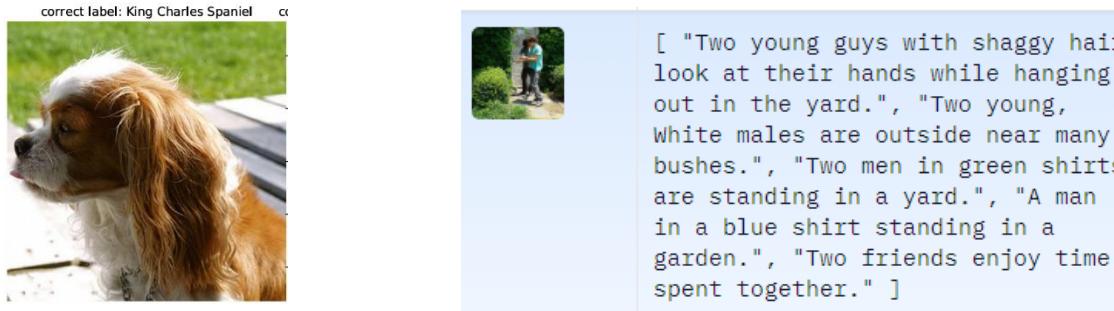
텍스트와 같은 외부 조건 정보가 U-Net 내의 크로스 어텐션을 통해 반영되어 생성 과정을 제어. 최종, 노이즈 제거된 잠재 표현을 다시 이미지로 복원해(디코더), 주어진 조건에 맞는 이미지 생성



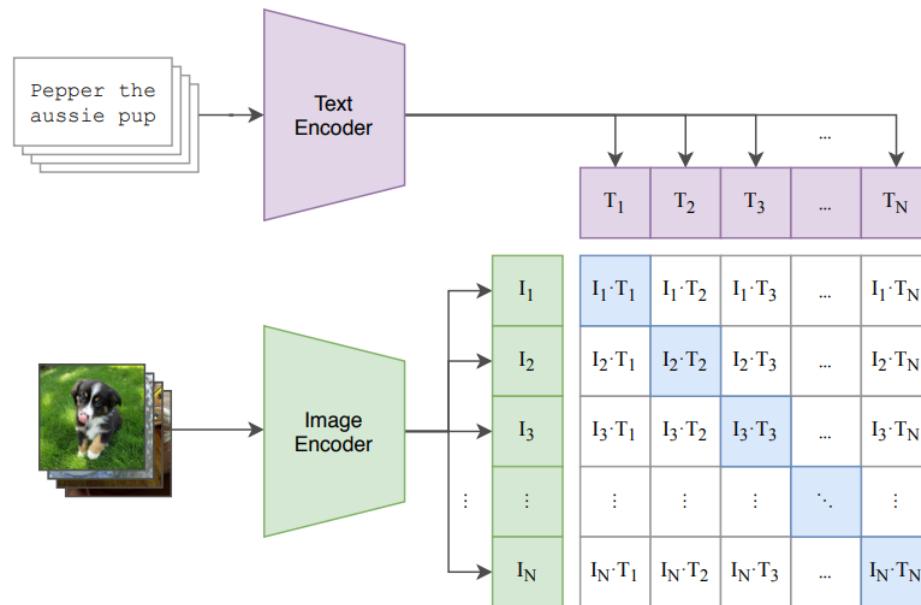
# 최신 AI 모델 왜 Transformer가 표준이 되었는가?

## CLIP (Contrastive Language–Image Pre-training)

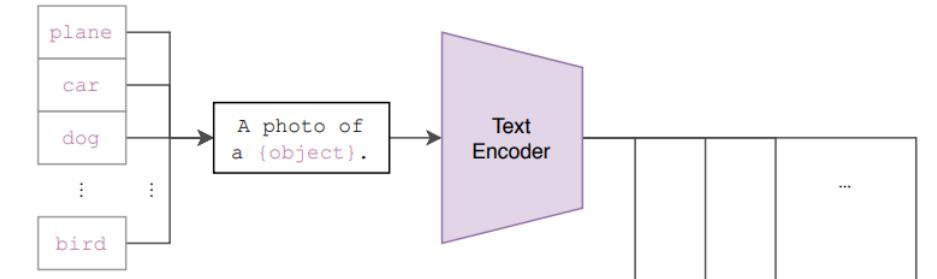
CLIP은 대규모의 (이미지, 텍스트) 쌍 데이터셋을 사용하여 대조 학습(Contrastive Learning) 방식으로 사전학습. 모델 목표는 이미지를 나타내는 임베딩과 해당 이미지를 설명하는 텍스트 임베딩을 공통된 잠재 공간으로 맵핑, 유사한 (이미지, 텍스트) 쌍은 서로 가깝게, 일치하지 않는 쌍은 멀리 떨어뜨리도록 학습하는 모델



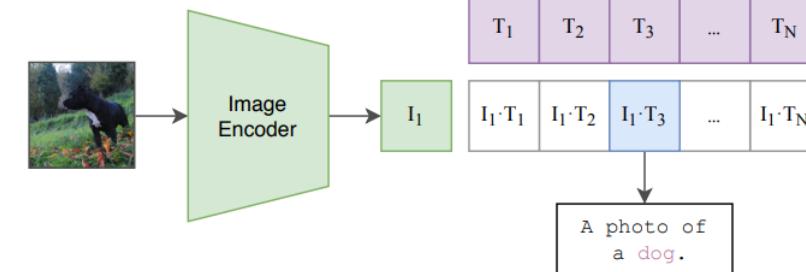
(1) Contrastive pre-training



(2) Create dataset classifier from label text



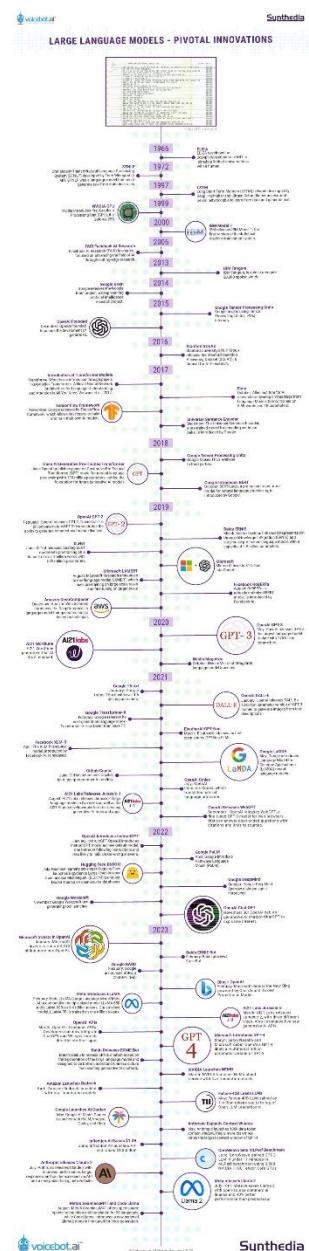
(3) Use for zero-shot prediction



# LLM 과 AI Agent

LLM과 RAG  
Ollama와 오픈소스모델  
AI Agent  
생성 AI와 Vibe Coding

# LLM 과 AI Agent LLM



## Introduction of Transformer Models

Transformer Models are introduced through papers like Googles Transformer: A Novel Neural Network Architecture for Language Understanding and Attention Is All You Need, Vaswani et al., 2017.



## Tensor Flow Framework

November: Google releases its TensorFlow framework, which allows developers to build and train their own AI models.

2017

$$y = f(w \cdot x + b)$$

Model	Parameters	Context Length
GPT-1	117M	512 tokens
GPT-4	175B - 1T (approx)	Up to 32,000 tokens
LLaMA 2	7B, 13B, 70B	4,096 tokens

$$\text{Memory (GB)} = \frac{\text{Parameters} \times \text{Bytes per Parameter}}{1024^3}$$

Substitute values:

$$\text{Memory (GB)} = \frac{175 \times 10^9 \times 2}{1024^3}$$

$$\text{Memory (GB)} = \frac{350 \times 10^9}{1,073,741,824} \approx 327 \text{ GB}$$

## OpenAI Chat GPT

November 30: OpenAI debuts generative AI chatbot ChatGPT to explosive interest.



2023



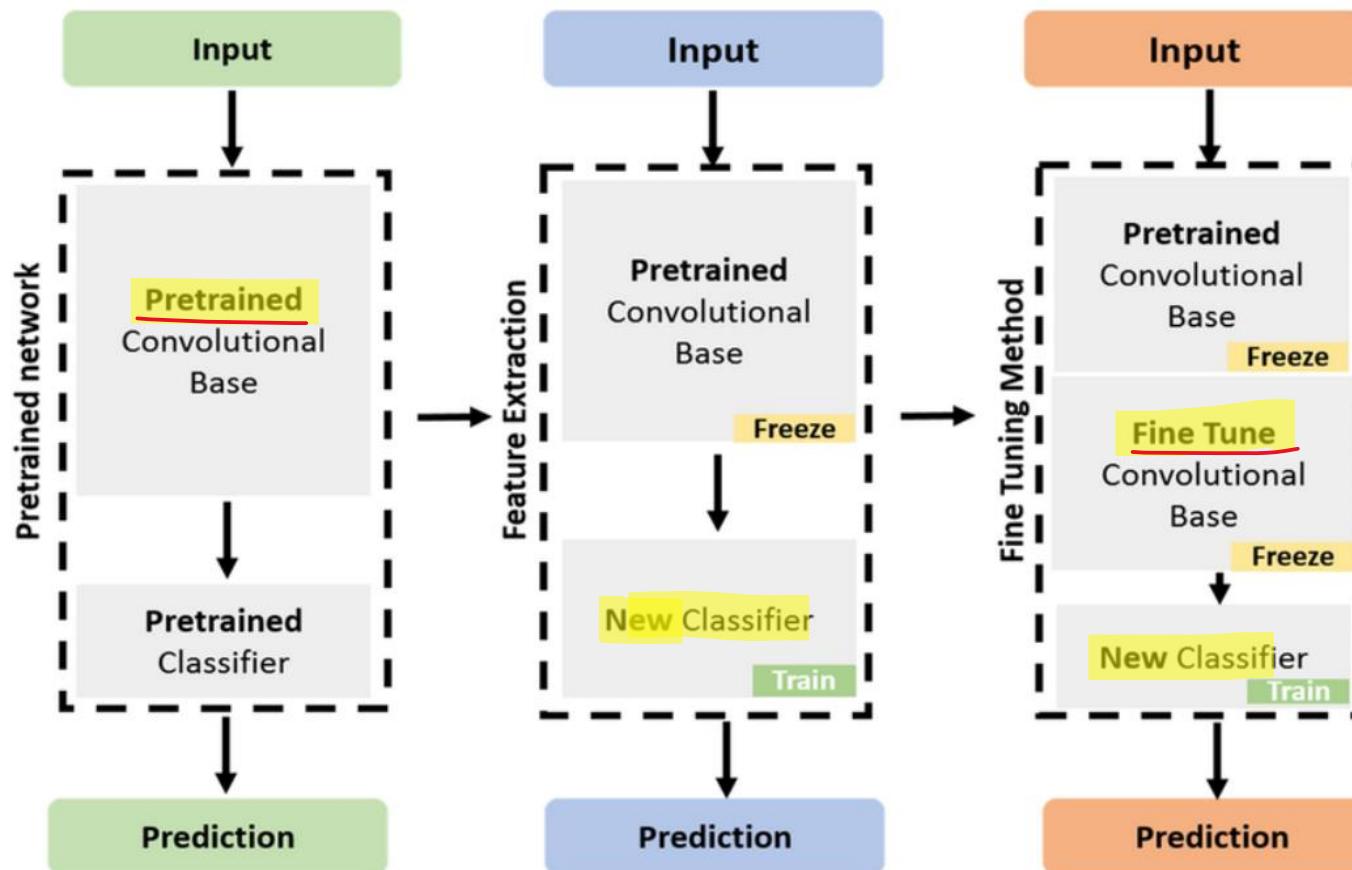
## Meta releases Llama 2

July 18th: Meta releases Llama 2 with open-source commercial license and 40% better performance than predecessor

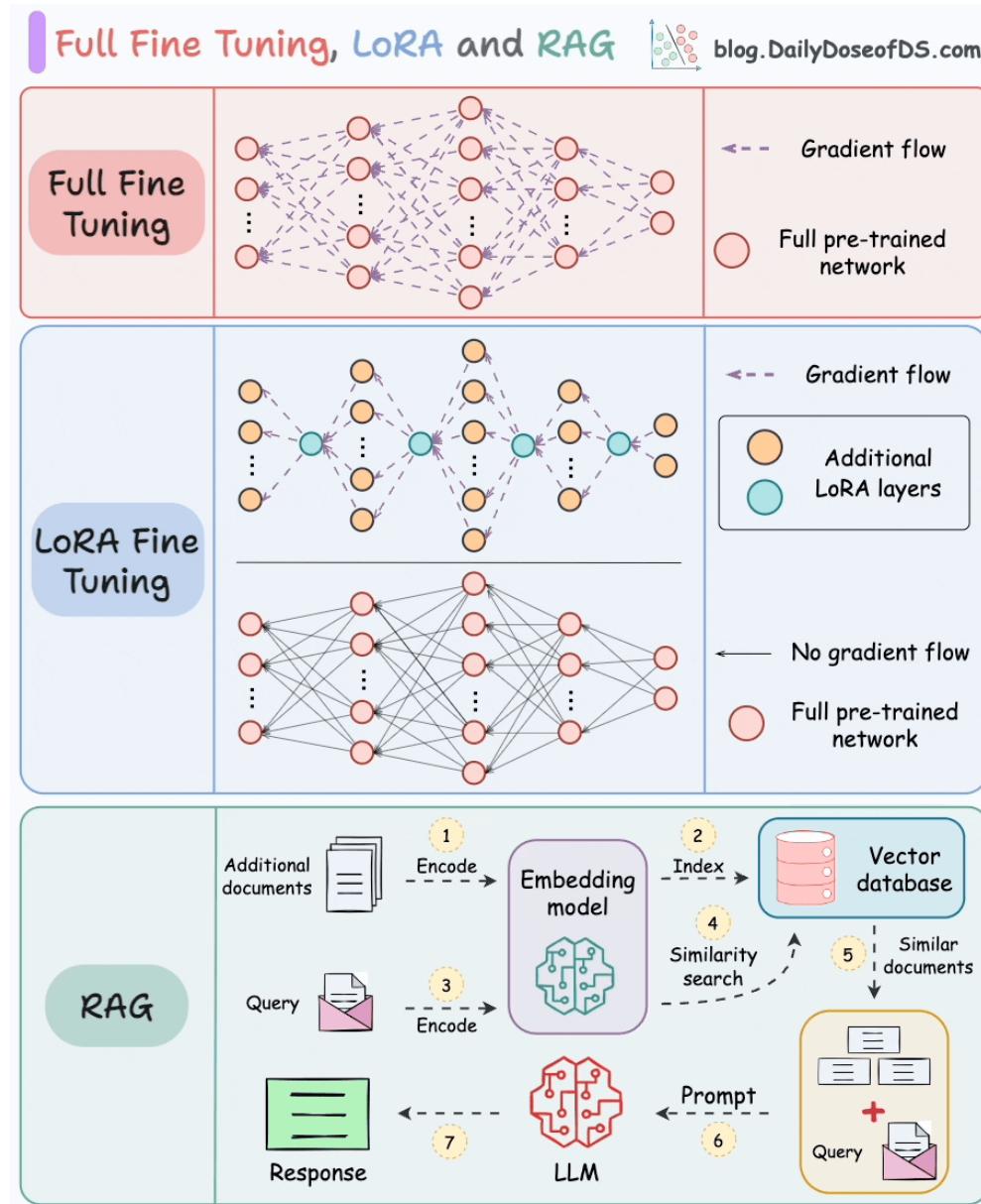
A Timeline of Large Language Model Innovation

# LLM 과 AI Agent LLM fine turning

대규모 언어 모델(LLM, Large Language Model)은 막대한 연산 자원을 요구하며, 전체 파라미터를 파인튜닝하는 방식은 메모리 사용량이 높고 계산 비용도 큼  
효율적으로 도메인 특화 파인튜닝을 수행할 수 있는 방식으로 LoRA(Low-Rank Adaptation)과 같은 PEFT(Parameter-Efficient Fine-Tuning) 이 사용되고 있음



# LLM 과 AI Agent LLM fine turning



$$z = f(W \cdot x + b)$$

$$W' = W + \Delta W$$

$$\Delta W = A \cdot B$$

Retrieval-Augmented Generation

Full-model Fine-tuning vs. LoRA vs. RAG - by Avi Chawla

# LLM 과 AI Agent LLM fine turning

## No fine tuning

```
[{"role": "user", "content": "Hello doctor, I have bad acne. How do I get rid of it?"]
```

Asking to truncate to max\_length but no maximum length is provided and the model has no predefined maximum length. Default to no truncation.

The first thing you need to do is to keep your face clean. Wash your face twice a day with a mild soap. Do not use any oil base make up. Use only water base make up. Do not use any oil base cream. Use only water base cream. Do not use any oil base perfume. Use only water base perfume. Do not use any oil base toothpaste. Use only water base toothpaste. Do not use any oil base shaving cream. Use only water base shaving cream. Do not use any oil base hair oil. Use only water base hair oil. Do not use any oil base hair cream

```
[{"role": "user", "content": "What are the profile properties of IfcStructuralCurveMemberVarying described by?"]]
```

What are the profile properties of IfcStructuralCurveMemberVarying described by? So...

The profile properties of IfcStructuralCurveMemberVarying

```
[{"role": "user", "content": "Hello doctor, I have bad acne. How do I get rid of it?"]]
```

Asking to truncate to max\_length but no maximum length is provided and the model has no predefined maximum length.

The first thing you need to do is to keep your face clean. Wash your face twice a day with a mild soap. Do not use m. Do not use any oil base perfume. Use only water base perfume. Do not use any oil base toothpaste. Use only water oil base hair oil. Use only water base hair oil. Do not use any oil base hair cream

```
[{"role": "user", "content": "What are the profile properties of IfcStructuralCurveMemberVarying described by?"]]
```

What are the profile properties of IfcStructuralCurveMemberVarying described by? So...

The profile properties of IfcStructuralCurveMemberVarying

# LLM 과 AI Agent LLM fine tuning

## After fine tuning

```
[{"role": "user", "content": "Hello doctor, I have bad acne. How do I get rid of it?"}]
```

Hello. Acne is a universal issue in the age group of 15 to 25 years due to hormonal changes in adolescents. However, there are multiple effective treatment options depending on the grade and severity of the problem. You can visit a dermatologist for a clinical evaluation and further treatment. If you are looking for home remedies, you can apply tea tree oil. It has antibacterial properties that help in reducing acne. You can apply 3 to 4 teaspoons of tea tree oil in 8 oz of water twice a day for three weeks. Additionally, you can apply honey with lemon juice twice a day. Lemon juice

```
[{"role": "user", "content": "What are the profile properties of IfcStructuralCurveMemberVarying described by?"}]
```

A start profile and an end profile (e.g. tapers) and the corresponding IfcStructuralCurveMemberConstants for linear structural elements. For non-linear structural elements, the profile properties are described by IfcStructuralCurveMemberHistory with an associated IfcStructuralCurveMemberConstants for linear structural elements only. For structural elements that represent reinforcements, the profile properties are described by IfcStructuralCurveMemberReactions with an associated IfcStructuralCurveMemberConstants for linear structural elements only. For structural elements that represent reinforcements, the profile properties are described by IfcStructuralCurveMemberReactions with an associated IfcStructuralCurveMemberConstants for linear structural elements only.

```
[{"role": "user", "content": "Hello doctor, I have bad acne. How do I get rid of it?"}]
```

Hello. Acne is a universal issue in the age group of 15 to 25 years due to hormonal changes in adolescents. However, there are multiple effective treatment options depending on the grade and severity of the problem. You can visit a dermatologist for a clinical evaluation and further treatment. If you are looking for home remedies, you can apply tea tree oil. It has antibacterial properties that help in reducing acne. You can apply 3 to 4 teaspoons of tea tree oil in 8 oz of water twice a day for three weeks. Additionally, you can apply honey with lemon juice

```
[{"role": "user", "content": "What are the profile properties of IfcStructuralCurveMemberVarying described by?"}]
```

A start profile and an end profile (e.g. tapers) and the corresponding IfcStructuralCurveMemberConstants for linear structural elements are described by IfcStructuralCurveMemberHistory with an associated IfcStructuralCurveMemberConstants for linear structural elements only. For structural elements that represent reinforcements, the profile properties are described by IfcStructuralCurveMemberReactions with an associated IfcStructuralCurveMemberConstants for linear structural elements only.

# LLM 과 AI Agent LLM fine turning

fine tuning. Epoch = 10 (over 5 hours)

```
[{"role": "user", "content": "Hello doctor, I have bad acne. How do I get rid of it?"]
```

Asking to truncate to `max_length` but no `maximum_length` is provided and the model has no predefined `maximum_length`.

Hello. Acne is a universal issue in the age group of 15 to 25 years due to hormonal changes in adolescents. However of the problem. Grade 1 to grade 3 can be treated conservatively, while grade 4 often requires oral antibiotics. Skin care products like Cetaphil and Clinique can be used at home. However, I suggest you consult a dermatologist to me know if I can assist you further. Regards, For more information consult a dermatologist online --> <https://bit.ly/Physician>, Dermatology Specialist, Skin Care Specialist, Acne Specialist, Cosmetic Dermatology, Anti-Aging Specialist, and Cosmetic injection specialist. For more information consult a dermatologist online --> <https://bit.ly/2023>. Please do not hesitate to ask further clarifications. Regards, For more information consult a dermatologist provided me.

```
[{"role": "user", "content": "What are the profile properties of IfcStructuralCurveMemberVarying described by?"]
```

A start profile and an end profile (e.g. tapers) and the corresponding `IfcStructuralCurveMemberEndProfile` and `IfcStructuralCurveMemberStartProfile`. In addition, a sweep operation is described by `IfcStructuralCurveMemberSweptAreaSolid`. For `IfcStructuralCurveMemberVarying` and `IfcStructuralCurveMemberVaryingSurface` pressures have to be defined as well as the tolerance for curve variations. Finally, the local axis system has to be defined too. For more information, see Löwner, M.O.; Gröger, G. In: Computer Aided Design and Applications, Vol. 10, No. 13, pp. 1313-1319. [CrossRef] [Web of Science (WoS)] [SCI] [e-EL] [Export to PDF] [Metadata] [BIMcert Handbuch] [BIMcert Database]

# LLM 과 AI Agent

## LLM fine turning

PEFT(Parameter-Efficient Fine-Tuning)는 사전 학습된 거대 언어 모델의 모든 가중치를 조정하지 않고, 특정 모듈 혹은 계층만을 학습 대상으로 설정함으로써 효율적인 파인튜닝을 가능하게 하는 접근 방식

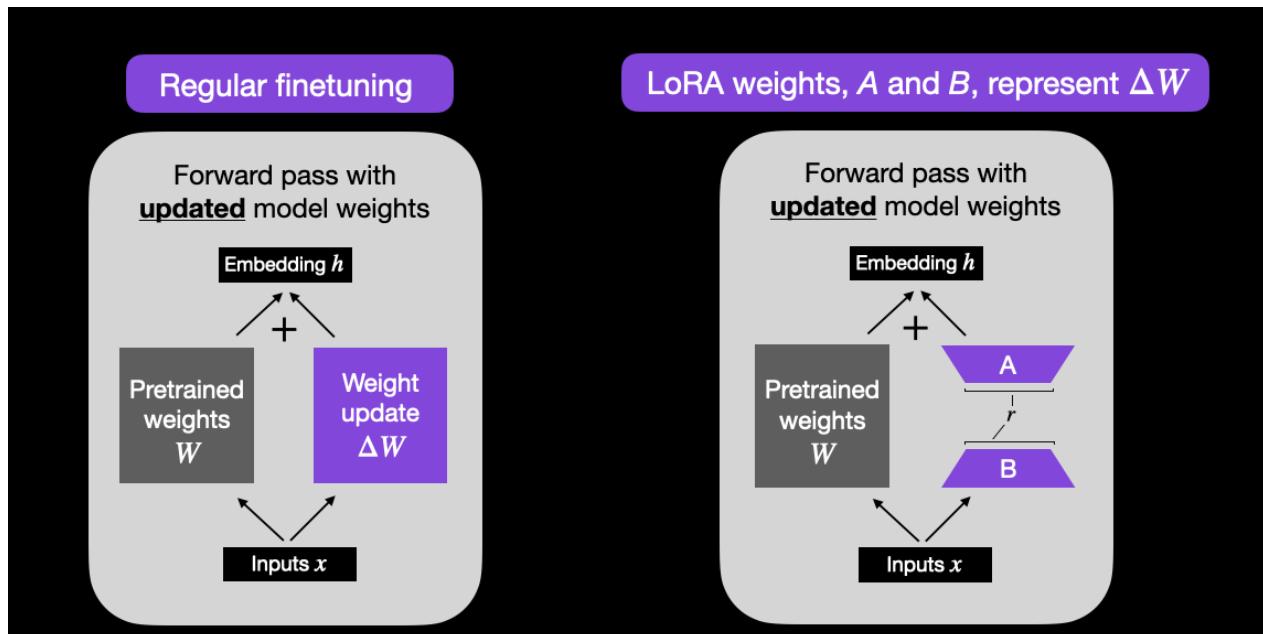
LoRA: 저랭크 행렬을 삽입하여 Linear 연산을 보완함.

Adapter Tuning: Transformer 블록에 작은 네트워크를 덧붙여 학습함.

Prefix/Prompt Tuning: 입력 시퀀스에 도메인 특화 토큰을 삽입함.

LoRA는 self-attention이나 feedforward 블록의 선형 계층에 대해 고정된 원본 가중치에 저랭크 행렬을 추가함으로써 파라미터 수를 줄이고 효율적인 학습을 가능

$$y = Wx \quad y = (W + BA)x$$



[Code LoRA from Scratch](#)

# LLM 과 AI Agent LLM fine turning

$$y = \mathbf{W}\mathbf{x} \quad y = (\mathbf{W} + \mathbf{B}\mathbf{A})\mathbf{x}$$

LoraConfig(

```
r=64,  
lora_alpha=32,  
target_modules=["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj"],  
lora_dropout=0.05,  
bias="none",  
task_type="CAUSAL_LM")
```

r: 저랭크 행렬의 차원, 작을수록 연산량 감소, 클수록 표현력 증가

lora\_alpha: 학습 중 scaling factor로 작동하며, 학습 안정성과 관련됨

target\_modules: LoRA가 삽입될 Transformer 내 Linear 레이어 이름

lora\_dropout: 학습 중 적용될 dropout 확률

bias: bias 항을 LoRA와 함께 학습할지 여부 (none, all, lora\_only 등)

task\_type: 파인튜닝 과제 종류 (CAUSAL\_LM, SEQ\_CLS 등). CAUSAL\_LM: 다음 토큰을 순차적으로 예측하는 언어 생성 모델 (예: GPT류)  
SEQ\_CLS: 입력 전체에 대해 단일 클래스를 예측하는 분류 모델 (예: 감정 분석, 문장 분류)

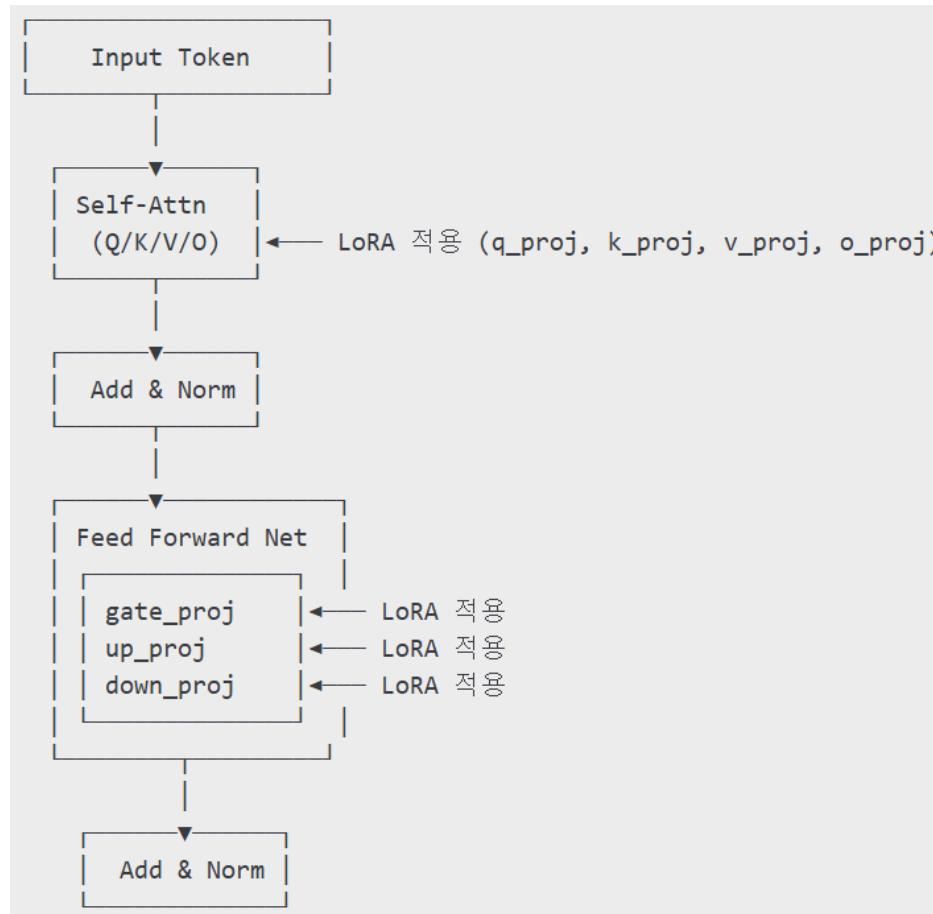
q\_proj, k\_proj, v\_proj, o\_proj: Self-Attention의 쿼리, 키, 밸류, 출력 projection에 대응함

- gate\_proj, up\_proj, down\_proj: Transformer 블록의 Feedforward Network(FFN)에서 사용되는 세 개의 핵심 선형 계층이다.
- gate\_proj: gating mechanism 적용 전 입력을 처리
- up\_proj: hidden dimension을 확장함 (ex. 4096 → 11008)
- down\_proj: 다시 차원을 줄이며 원래 출력으로 복원

# LLM 과 AI Agent LLM fine turning

q\_proj, k\_proj, v\_proj, o\_proj: Self-Attention의 쿼리, 키, 밸류, 출력 projection에 대응함

- gate\_proj, up\_proj, down\_proj: Transformer 블록의 Feedforward Network(FFN)에서 사용되는 세 개의 핵심 선형 계층이다.
- gate\_proj: gating mechanism 적용 전 입력을 처리
- up\_proj: hidden dimension을 확장함 (ex. 4096 → 11008)
- down\_proj: 다시 차원을 줄이며 원래 출력으로 복원



# LLM 과 AI Agent 양자화 (Quantization)

양자화는 모델 파라미터를 float 대신 int4/8로 표현하여 자원 사용을 줄이고 계산속도를 향상시키는 기술

bitsandbytes는 Hugging Face에서 채택된 양자화 라이브러리이며, 특히 4bit 양자화에서 매우 효과적

```
BitsAndBytesConfig(  
    load_in_4bit=True,  
    bnb_4bit_use_double_quant=True,  
    bnb_4bit_quant_type="nf4",  
    bnb_4bit_compute_dtype=torch.bfloat16  
)
```

load\_in\_4bit: 모델을 4bit로 로드할지 여부

bnb\_4bit\_use\_double\_quant: 이중 양자화 수행으로 표현력 향상

bnb\_4bit\_quant\_type:

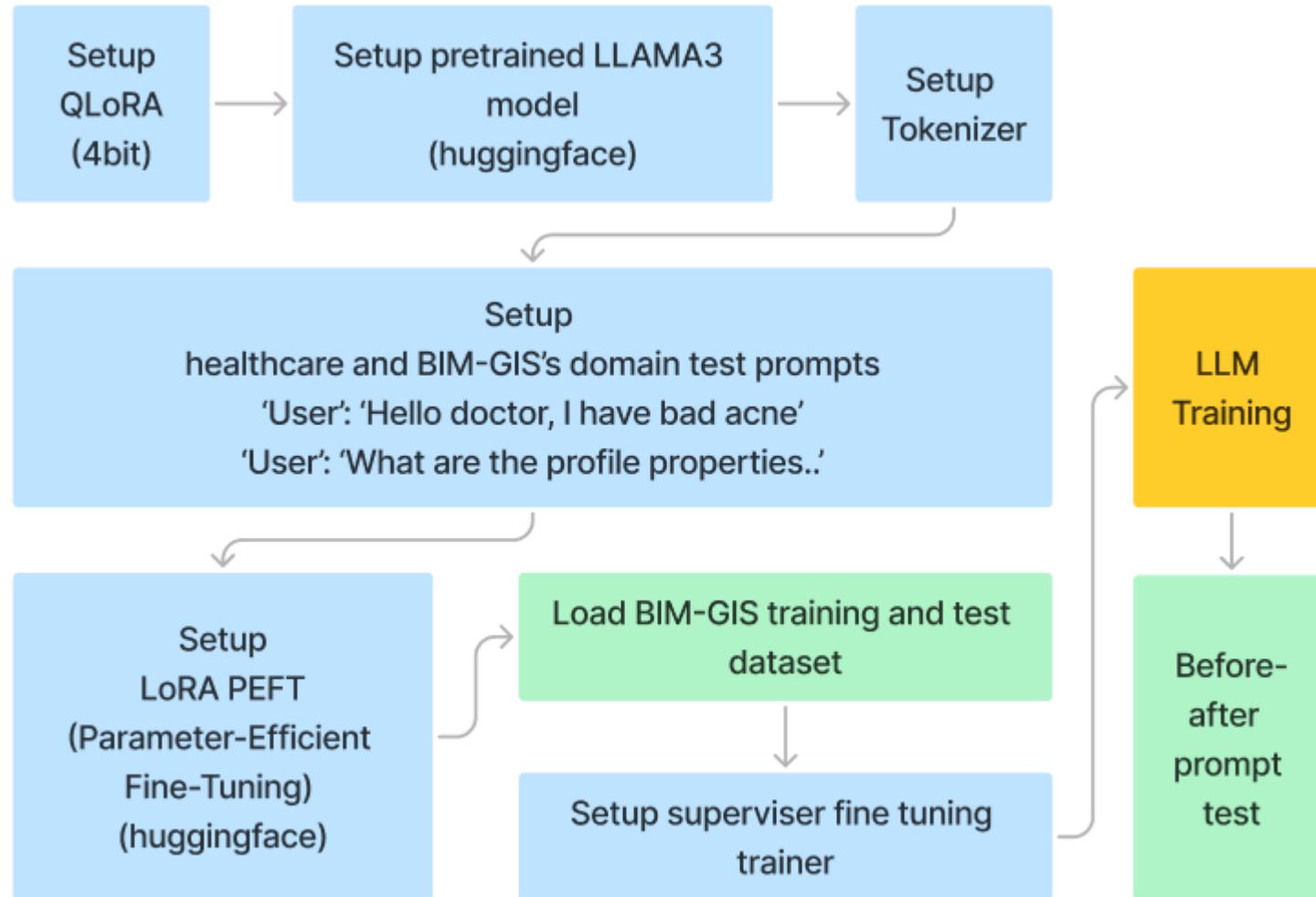
nf4: NormalFloat4, 정규화된 4비트 부동소수점

fp4: 기본 부동소수점 4비트 양자화, 정밀도는 낮음

bnb\_4bit\_compute\_dtype: 계산 dtype (bfloat16, float16, float32) 지정. 일반적으로 bfloat16 사용

# LLM 과 AI Agent

## LLM fine turning



```
{'loss': 2.6513, 'grad_norm': 1.0972543954849243, 'learning_rate': 0.00019908787541713017, 'epoch': 0.11}  
{'loss': 2.3663, 'grad_norm': 1.041242241859436, 'learning_rate': 0.00019906562847608455, 'epoch': 0.12}  
{'loss': 2.4729, 'grad_norm': 1.1358354091644287, 'learning_rate': 0.00019904338153503895, 'epoch': 0.12}  
{'loss': 2.3378, 'grad_norm': 1.1035994291305542, 'learning_rate': 0.00019902113459399333, 'epoch': 0.12}  
{'loss': 2.3338, 'grad_norm': 1.300874948501587, 'learning_rate': 0.0001989988876529477, 'epoch': 0.12}...{'loss': 2.3338, 'grad_norm': 1.300874948501587, 'learning_rate': 0.0001989988876529477, 'epoch': 0.12}
```

# LLM 과 AI Agent

## LLM fine turning

### Pretraining

- ▶ **Dataset:**
  - ▶ Raw text
  - ▶ Few trillions of tokens
- ▶ **Task:**
  - ▶ Next word prediction
- ▶ **Compute:**
  - ▶  $O(1-10K)$  GPUs
  - ▶ Weeks/months of training

### Instruction-Tuning

- ▶ **Dataset:**
  - ▶ Paired: (Prompt, Response)
  - ▶  $O(10-100K)$  instructions
- ▶ **Task:**
  - ▶ Next word prediction (Masked)
- ▶ **Compute:**
  - ▶  $O(1-100)$  GPUs
  - ▶ Few hrs/days

### Learning from Human Feedback

- ▶ **Dataset:**
  - ▶ Human Preference Data
  - ▶  $O(10-100K)$
- ▶ **Task:**
  - ▶ RLHF/DPO
- ▶ **Compute:**
  - ▶  $O(1-100)$  GPUs
  - ▶ Few hrs/days

# LLM 과 AI Agent

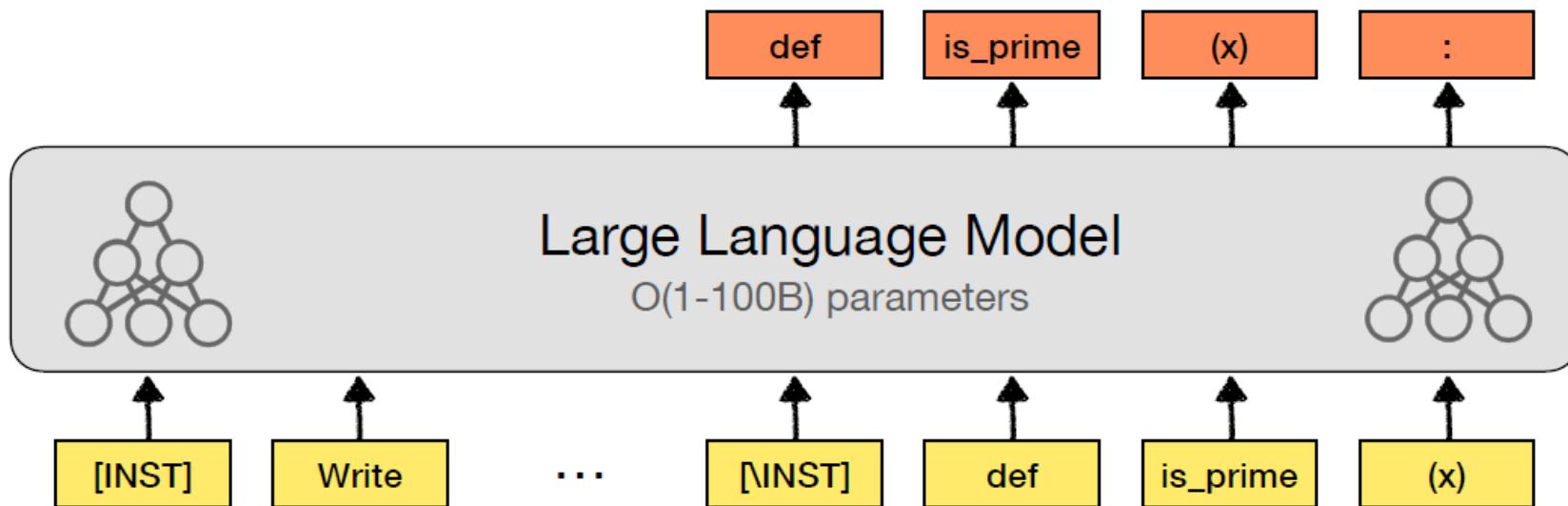
LLM fine turning

Prompt:

```
[INST] Write a python function to find whether the input number is prime. [INST]
```

```
def is_prime(x: int):
    """
    takes as input an integer x. Returns True if x is prime and False otherwise
    """
    if x <= 1:
        return False
    for i in range(2, int(x ** 0.5) + 1):
        if x % i == 0:
            return False
    return True
```

Response:



# LLM 과 AI Agent LLM fine turning

🔍 Expand to see the descriptions of 34 models

Model A

Write a python function to find whether the input number is prime.

Here is a Python function to check if a given number is prime:

```
def is_prime(num):
    if num < 2:
        return False
    for i in range(2, int(num ** 0.5) + 1):
        if num % i == 0:
            return False
    return True

# Test the function
number = int(input("Enter a number: "))
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.)
```

You can use this function to determine whether a given number is prime or not.

Model B

Write a python function to find whether the input number is prime.

Sure, let's break down the problem into steps and then I'll write the Python function for you.

A number is considered prime if it has exactly two factors: 1 and itself. All numbers greater than 1 have at least three factors, one of which is 1, so we can start our range from 2.

Here is a Python function that checks whether a number is prime:

```
def is_prime(number):
    # We start from 2 because 1 and the number itself are always factors
    for i in range(2, int(number**0.5) + 1):
        # If we find a factor, the number is not prime
        if number % i == 0:
            return False

    # If no factors are found, the number is prime
    return True
```

This function takes an integer as input and returns a boolean value. It starts by checking if the number is less than or equal to 1. If so, it immediately returns `True` for 2 and `False` for all others because these are the only prime numbers below 12.

If the number is greater than 1, the function then checks for factors from 2 up to the square root

👉 A is better

👉 B is better

👉 Tie

👉 Both are bad

👉 Enter your prompt and press ENTER

Send

New Round

Regenerate

Share

# LLM 과 AI Agent

AI Agent

## 환각 문제



Why wasn't that answer 35 as you predicted?



I apologize for my mistake earlier. The correct set of operations to arrive at 35 is to double the number 8, which is 16, subtract 3 from it to get 13, multiply it by 5 to get 65, subtract 30 to get 35, and then divide it by 2 to get the final answer of 17.

# LLM 과 AI Agent RAG

## RAG(Retrieval-Augmented Generation)

대규모 언어 모델(LLM: Large Language Model)의 한계를 극복하기 위해 외부 지식 기반을 검색해 생성 과정에 반영하는 방식

RAG는 외부 문서나 데이터베이스에서 관련 정보를 검색(Retrieval)하고, 이를 입력에 증강(Augmentation)하여, 최종적으로 텍스트를 생성(Generation)하는 세 단계로 구성

RAG는 2020 년 [Facebook AI Research\(FAIR\)](#)에

제출

### Information Retrieval



### Text Generation



Close-book exam  
(Hard mode)



### Retrieval-Augmented Text Generation



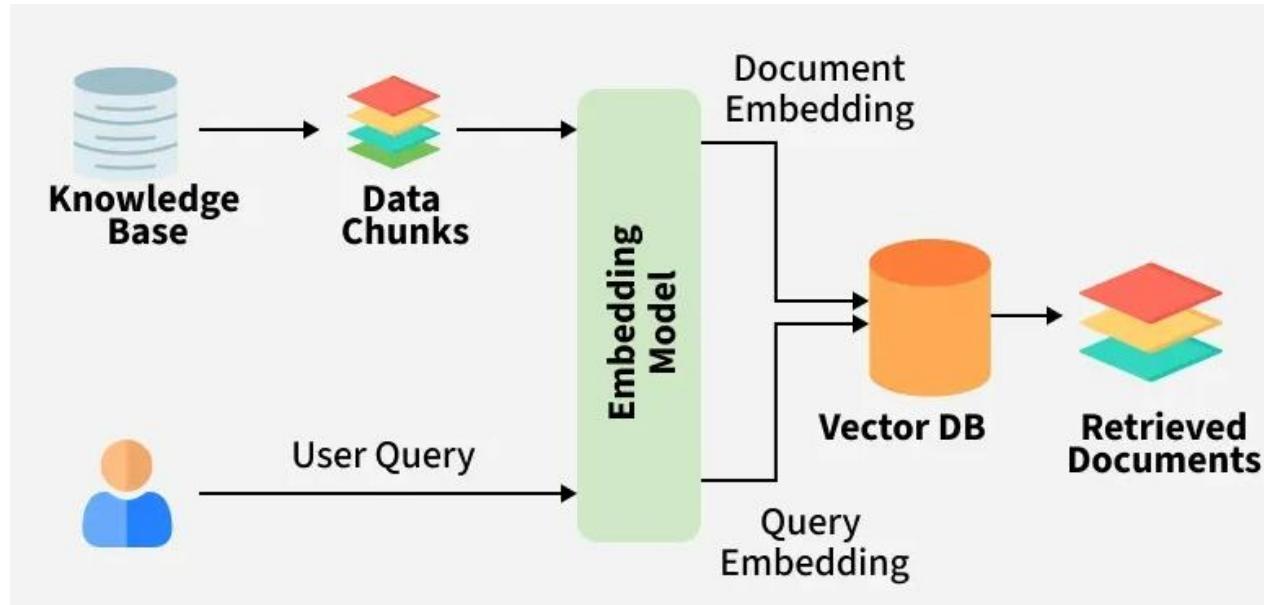
Open-book exam  
(Easy mode)



# LLM 과 AI Agent RAG

## 검색

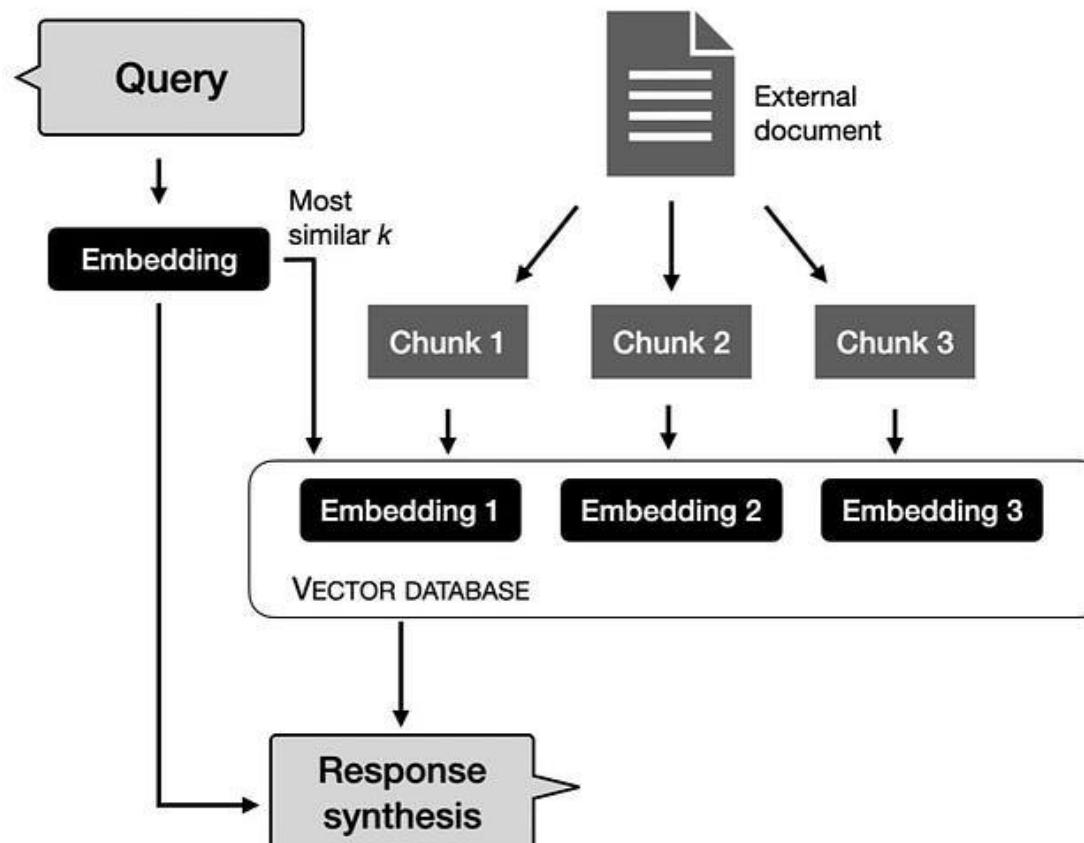
질의 문장을 벡터데이터베이스에 저장된 청킹된 문서 임베딩 벡터와 유사도 계산을 통해 질의에 관련된 문서들을 검색함



# LLM 과 AI Agent RAG

## 청킹(chunking)

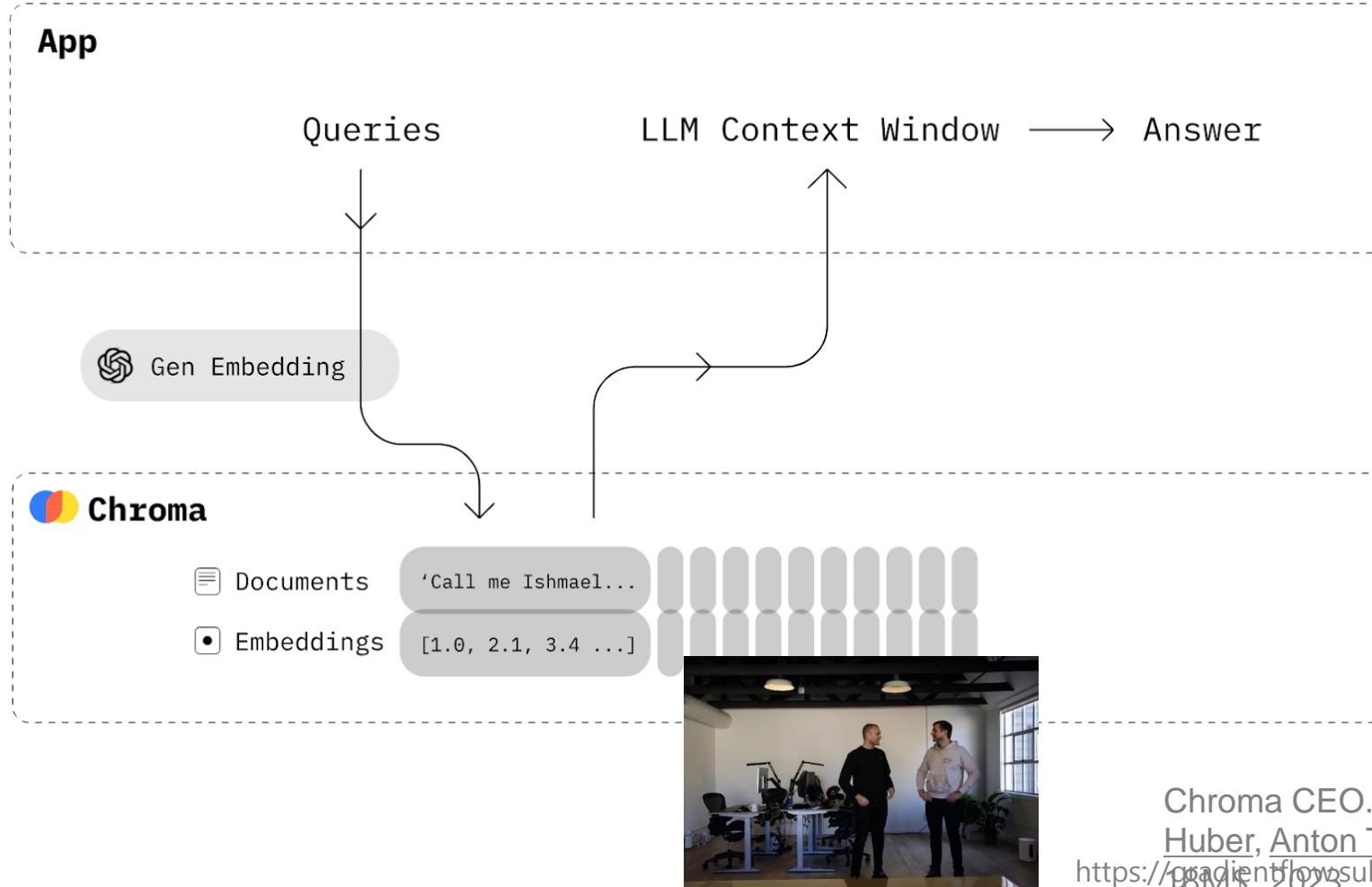
대규모 텍스트 데이터(예: 긴 문서, 책, 웹페이지 모음)를 의미론적으로 정리 요약되어 있고, 컨텍스트 윈도우(context window) 내 적절한 크기를 가지는 독립적인 단위인 '청크'들로 분할하는 전처리 과정



# LLM 과 AI Agent RAG

## 증강

검색된 문서를 사용자의 원 질문과 결합하여 LLM 입력 프롬프트를 구성



Chroma CEO. Jeff

Huber, Anton Troynikov

<https://gradientflowsubstack.com/p/best-practices-in-retrieval-augmented-lm>

# LLM 과 AI Agent RAG

## 증강

RAG 시스템에서 문서를 검색 가능하게 만들기 위해서는 먼저 문서를 처리 가능한 단위로 나눈 뒤, 각 단위를 벡터로 변환.

- LLM이나 임베딩 모델은 입력 토큰 수에 제한이 있으며, 긴 문서를 한 번에 처리할 수 없음
- 의미 있는 문단 또는 문장 단위로 분할하면 정보가 손실 방지. 검색 품질 개선
- 검색 단계에서 세부적인 단위(챕터)로 비교함으로써, 질문과 가장 관련 있는 문맥을 찾아낼 수 있음

증강 생성 시 과도한 정보가 입력되면 노이즈가 발생할 수 있으므로, 적절한 단위로 잘라 사용하는 것이 성능 향상에 기여

# 청킹

```
chunks = split_document(doc, chunk_size=500,  
overlap=100)
```

# 임베딩

```
for chunk in chunks:  
    v = embed_model(chunk)  
    vector_store.append(v)
```

# LLM 과 AI Agent RAG

## 증강 후 생성

롬프트를 LLM(OpenAI GPT-4, Claude, LLaMA 등)에 전달하여 응답을 출력

1. Augmented Prompt를 형성한다 (사용자 질문 + 검색된 문서).
2. 해당 Prompt를 LLM의 입력으로 전달한다.
3. LLM은 사전 학습된 확률 기반 언어 모델링을 통해 다음 토큰을 반복 생성하며 응답을 생성한다.
4. 필요 시, temperature, top-k, top-p 같은 샘플링 파라미터를 조정하여 생성 다양성과 품질을 조절한다.

You are a helpful assistant. Use the following context to answer the question.

Context:

{retrieved\_context}

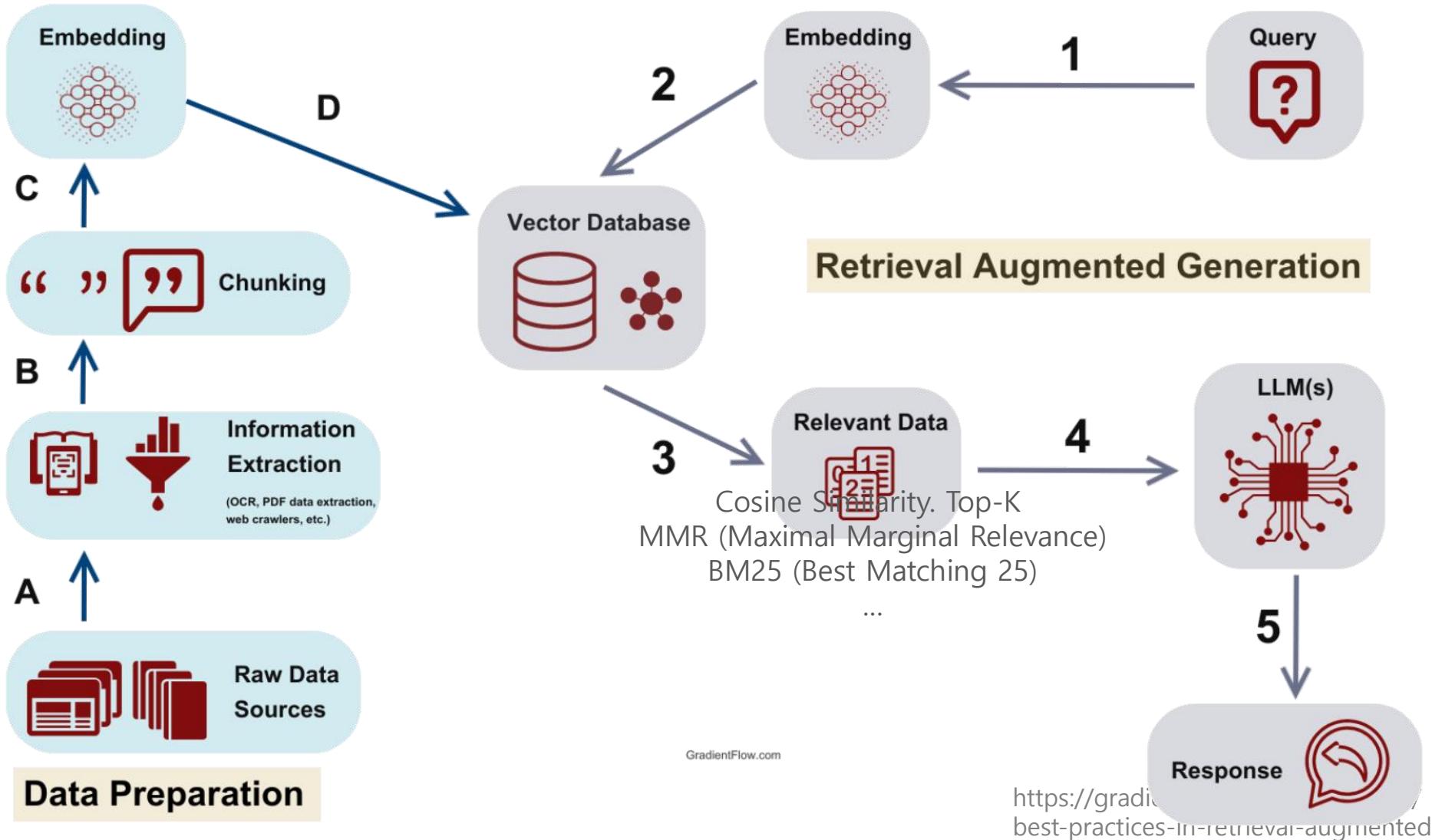
Question:

{user\_question}

Answer:

# LLM 과 AI Agent RAG

## 전체 RAG 흐름



# LLM 과 AI Agent RAG

## 에이전트 개발 도구

도구 이름	설명	링크
AutoGPT	텍스트 기반 입력을 통해 자동으로 여러 작업을 수행하는 AI 에이전트. GPT 모델을 활용하여 다양한 작업을 자동화할 수 있도록 설계됨.	<a href="#">AutoGPT</a> <a href="#">GitHub</a>
Pydantic	AI 에이전트 및 모델 구축을 위한 데이터 검증 및 설정 도구. Pydantic을 사용하여 AI 모델의 입력과 출력을 구조적으로 관리하고 검증할 수 있다.	<a href="#">PydanticAI</a>
LlamaIndex	Llama 모델을 활용하여 효율적인 데이터 검색 및 에이전트 기능을 구현하는 도구. 특히 대규모 텍스트 데이터 처리 및 검색에 강점.	<a href="#">LlamaIndex</a> <a href="#">GitHub</a>
CrewAI	협업을 중심으로 AI 에이전트를 관리하는 도구. 여러 AI 에이전트를 연결하여 복잡한 문제를 해결하는데 유용.	<a href="#">CrewAI</a> <a href="#">Website</a>
LangGraph	그래프 데이터베이스 기반의 AI 에이전트 관리 도구로, 복잡한 데이터 관계를 효과적으로 처리하고 탐색할 수 있도록 돋는다.	<a href="#">LangGraph</a> <a href="#">GitHub</a>

# LLM 과 AI Agent RAG

# PDF 읽기 및 청킹

```
from langchain.document_loaders import PyPDFLoader
from langchain.text_splitter import RecursiveCharacterTextSplitter
loader = PyPDFLoader("my_doc.pdf")
docs = loader.load()
splitter = RecursiveCharacterTextSplitter(chunk_size=500, chunk_overlap=100)
chunks = splitter.split_documents(docs)
```

# 임베딩 및 벡터 저장

```
from langchain.vectorstores import FAISS
from langchain.embeddings import OpenAIEmbeddings
embedding_model = OpenAIEmbeddings()
vectorstore = FAISS.from_documents(chunks, embedding_model)
```

# 질문-응답 체인 구성

```
from langchain.chains import RetrievalQA
from langchain.chat_models import ChatOpenAI
retriever = vectorstore.as_retriever(search_type="mmr")
qa_chain = RetrievalQA.from_chain_type(
    llm=ChatOpenAI(),
    chain_type="stuff",
    retriever=retriever
)
```

# 질문에 대한 응답 생성

```
query = "이 문서에서 저자의 주요 주장은 무엇인가요?"
result = qa_chain.run(query)
print(result)
```

# LLM 과 AI Agent RAG

## 예시



# LLM 과 AI Agent RAG

## 예시

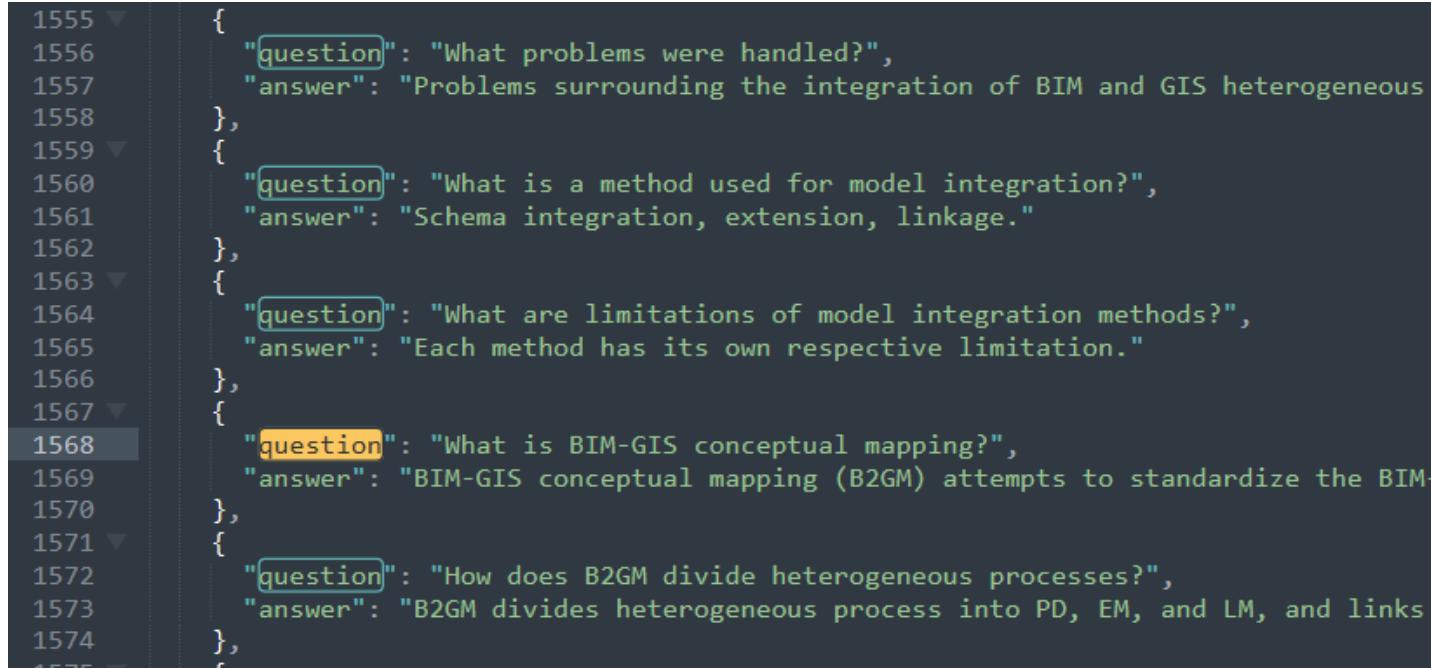
```
client = OpenAI(  
    base_url = 'http://localhost:11434/v1',  
    api_key='ollama',  
)  
  
template = {  
    "question": " ",  
    "answer": " "  
}
```

```
def generate_questions_answers(text_chunk):  
    messages = [  
        {'role': 'system', 'content': 'You are an API that converts bodies of text  
            into multiple questions and answers into a JSON format. Each JSON  
            should contain a single question with a single answer. Only respond  
            with the JSON and no additional text. \n.'},  
        {'role': 'user', 'content': 'Text: ' + text_chunk}  
    ]  
  
    response = client.chat.completions.create(  
        model='llama3',  
        messages=messages,  
        max_tokens=2048,  
        n=1,  
        stop=None,  
        temperature=0.7,  
    )
```

# LLM 과 AI Agent RAG

```
{  
    "question": "What is the URL to access information on jetgeo?",  
    "answer": ".com/jetgeo/IFC2GML"  
},  
{  
    "question": "When was the ISO TC 211 introduction to UML accessed?",  
    "answer": "20 March 2020"  
},  
{  
    "question": "Who wrote Learning UML 2.0 and what is its edition?",  
    "answer": "Miles, R.R.; Hamilton, K., 1st ed."  
},
```

```
1555 ▾ {  
1556     "question": "What problems were handled?",  
1557     "answer": "Problems surrounding the integration of BIM and GIS heterogeneous m  
1558 },  
1559 ▾ {  
1560     "question": "What is a method used for model integration?",  
1561     "answer": "Schema integration, extension, linkage."  
1562 },  
1563 ▾ {  
1564     "question": "What are limitations of model integration methods?",  
1565     "answer": "Each method has its own respective limitation."  
1566 },  
1567 ▾ {  
1568     "question": "What is BIM-GIS conceptual mapping?",  
1569     "answer": "BIM-GIS conceptual mapping (B2GM) attempts to standardize the BIM-G  
1570 },  
1571 ▾ {  
1572     "question": "How does B2GM divide heterogeneous processes?",  
1573     "answer": "B2GM divides heterogeneous process into PD, EM, and LM, and links a  
1574 },  
1575 }
```



# LLM 과 AI Agent RAG

Open Access Article

## Development of a Conceptual Mapping Standard to Link Building and Geospatial Information

by Taewook Kang

Korea Institute of Civil Engineering and Building Technology, Ilsanseo-Gu Goyang-Si 10223, Korea

ISPRS Int. J. Geo-Inf. 2018, 7(5), 162; <https://doi.org/10.3390/ijgi7050162>

Submission received: 18 February 2018 / Revised: 7 April 2018 / Accepted: 19 April 2018 /

Published: 24 April 2018

(This article belongs to the Special Issue Building Information Modeling and 3D GIS Integration: From the Theoretical to the Practical)

Download

Browse Figures

Versions Notes

### Abstract

This study introduces the BIM (building information modeling)-GIS (geographic information system) conceptual mapping (B2GM) standard ISO N19166 and proposes a mapping mechanism. In addition, the major issues concerning BIM-GIS integration, and the considerations that it requires, are discussed. The B2GM is currently being standardized by the spatial-information international standardization organization TC211. Previous studies on BIM-GIS integration seem to focus on the integration of different types of model schemas and on the implementation of service interfaces. B2GM concerns the clear definition of the conditions and methods for mapping the object information required from the user's use-case viewpoint for city-scale mapping. The benefits of the B2GM approach are that the user is able directly control the BIM-GIS linkage and integration process in order to acquire the necessary objection information. This can reveal cases of possibly unclear BIM-GIS integration outside the black box in an explicit and standard way, so that the user can distinctively predict the final output obtainable from the BIM-GIS integration. This study examined B2GM in terms of its development background, components, and several utilization examples, as well as the levels and considerations of the integration of different BIM-GIS models.

Keywords: BIM; GIS; conceptual mapping; ISO 19166; BG-IL

Version: 551.61		CUDA Version: 12.4	
Bus-Id	Disp.A	Volatile Uncorr.	ECC
Memory-Usage		GPU-Uti l	Compute M.
		MIG M.	
00000000:01:00.0	On		N/A
User 5074MIB /	8192MIB	86%	Default
			N/A

<https://daddynkidsmakers.blogspot.com/2024/02/github-copilot-ai.html>

Drag and drop files here  
Limit 200MB per file • PDF

[Browse files](#)

GeoBIM.pdf 3.6MB

X

### Chat

what is BIM? 

BIM (Building Information Modeling) is a method used in the construction industry to create and manage digital representations of physical and functional characteristics of places. It includes defining object information such as runtime, geometry, property, and relationship for building components like walls, doors, and rooms. The geometry information uses boundary representation with support for vertices, edges, loops, and faces. Property information categorizes attributes and defines name, value type, and initial value pairs. System properties, including the BIM element name and GUID, are automatically created when making a model. Relationships between elements are also defined, following UML relationships.

what is use cases of BIM? 

BIM-GIS integration is used for scaling building information modeling (BIM) to city level use cases. This method includes a BIM-GIS element, geometry, and property set mapping with external datasets. The goal is to enable flexible information mapping and integration, focusing on user's specific requirements. Previous studies suggest various methods for BIM-GIS integration, but this study proposes using the B2GM standard under ISO/TC211.

tell me about BIM-GIS standard? 

The BIM-GIS standard, also known as Building Information Modeling-Geographic Information System, is a method for

# LLM 과 AI Agent Ollama 와 오픈소스모델



ollama run llama3.2-vision  
what is this figure? ./zaha\_ddp.jpg

```
>>> what is this figure? ./zaha_ddp.jpg
```

Added image './zaha\_ddp.jpg'

The image appears to be an interior view of a large, modern building with white architecture seems futuristic and minimalist, with clean lines and curves.

Upon closer inspection, it's possible that the image depicts a public space or building, such as a lobby, atrium, or hallway. The overall aesthetic suggests design, possibly intended to evoke feelings of calmness and serenity in those

이미지는 흰색 벽과 바닥이 있는 크고 현대적인 건물의 내부를 보여주는 것처럼 보입니다. 건축물은 미래적이고 미니멀하며, 선과 곡선이 깔끔합니다.

자세히 살펴보면 이미지가 로비, 아트리움 또는 복도와 같이 건물 내부의 공공 공간이나 상업 구역을 묘사한 것일 수 있습니다. 전반적인 미학은 현대적이고 세련된 디자인을 시사하는데, 아마도 공간에 들어오는 사람들에게 차분함과 평온함을 불러일으키기 위한 의도일 것입니다.

이미지는 흰색 벽과 바닥이 있는 크고 열린 공간을 보여줍니다. 천장도 흰색이고 천장 가장자리를 따라 여러 줄의 리세스 조명이 있습니다. 방 중앙에는 공간을 두 구역으로 나누는 길고 곡선형 벽이 있습니다. 벽에는 길이를 따라 일련의 창문이 있어 자연광이 방으로 들어올 수 있습니다.

곡선 벽 앞에는 반짝이는 흰색 바닥이 있는 넓고 열린 공간이 있습니다. 이곳은 건물 내의 어떤 종류의 공공 공간이나 상업 공간인 듯합니다. 이미지에는 사람이 보이지 않지만 사람들이 모이거나 모이는 장소일 가능성이 높습니다.

이미지의 전반적인 분위기는 현대적이고 매끈합니다. 깔끔한 선, 곡선, 미니멀리즘의 미학은 현대적인 디자인 스타일을 시사합니다. 기본 색상으로 흰색을 사용하면 밝기와 개방감이 생기며, 이는 공간에 들어오는 사람들에게 차분함과 평온함을 불러일으키려는 의도일 수 있습니다.

# LLM 과 AI Agent Ollama 와 오픈소스모델

## Ollama

로컬에서 대규모 언어 모델(LLM)을 실행 활용할 수 있도록 도와주는 경량화된 도구

GPT, LLaMA, Mistral 등의 오픈소스 모델을 로컬 환경에서 직접 불러와 프롬프트 응답이나 RAG 검색 등에 사용할 수 있도록 설계

Docker 없이도 작동하며, LangChain과의 통합도 매우 용이

- <https://ollama.com> 에 접속
- 운영체제(Windows)를 선택하고 설치 프로그램(.exe)을 다운로드
- 설치 후, 커맨드 프롬프트(CMD)나 PowerShell을 열어 ollama 명령어가 인식되는지 확인

```
F:\projects>ollama list
NAME           ID      SIZE    MODIFIED
tinyllama:latest 2644915ede35  637 MB  8 days ago
llama3:8b-instruct-q4_K_M 9b8f3f3385bf  4.9 GB  8 days ago
qwen2.5-coder:7b   dae161e27b0e  4.7 GB  3 weeks ago
codegemma:7b      0c96700aaada  5.0 GB  3 weeks ago
gemma3:latest    a2af6cc3eb7f  3.3 GB  5 weeks ago
llama3.2-vision:latest 38107a0cd119  7.9 GB  8 months ago | 확인

F:\projects>ollama run gemma3
>>> 안녕
안녕하세요! 무엇을 도와드릴까요? 😊

>>> 너는 누구?
저는 Google에서 개발한 대규모 언어 모델입니다.

쉽게 말하면, 방대한 양의 텍스트 데이터를 학습하여 인간처럼 대화하고, 질문에 답하고, 텍스트를 생성할 수 있는 인공지능입니다.
```

# LLM 과 AI Agent Ollama 와 오픈소스모델

Web 기반 Ollama 서비스 구동

다음 링크에서 올라마를 설치

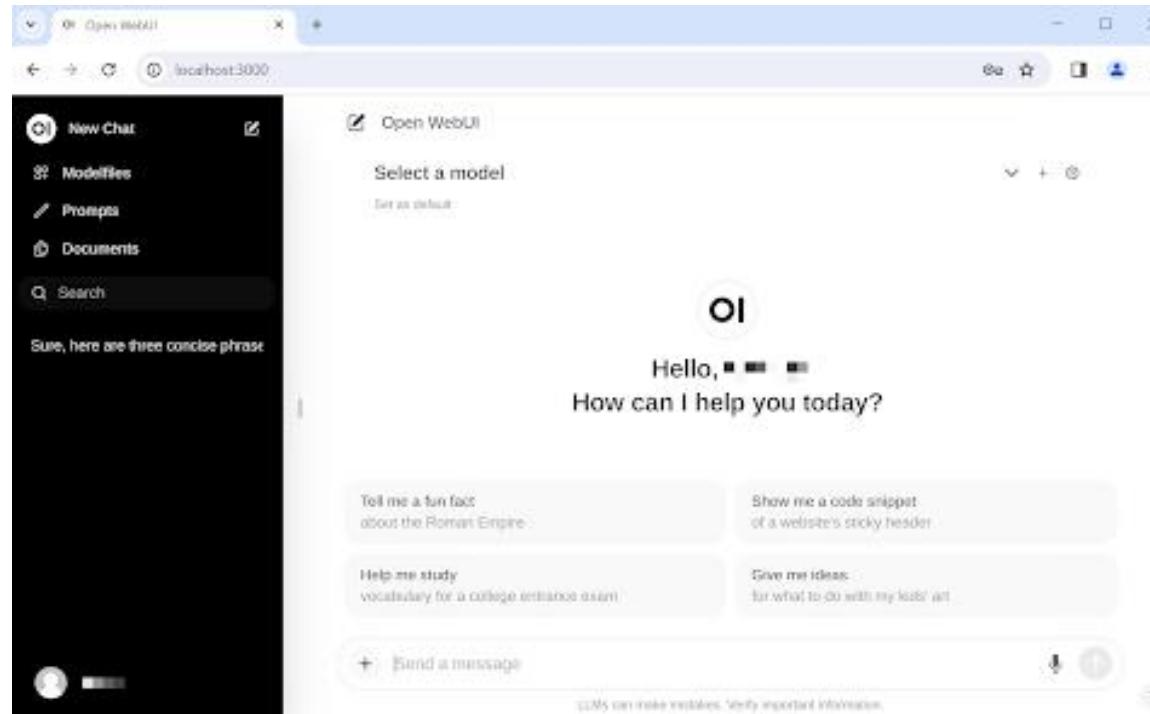
- [Getting Started | Open WebUI](#)

웹에서 실행하기 위해 다음 링크 참고해 도커를 설치. 재부팅

- [Windows | Docker Docs](#)

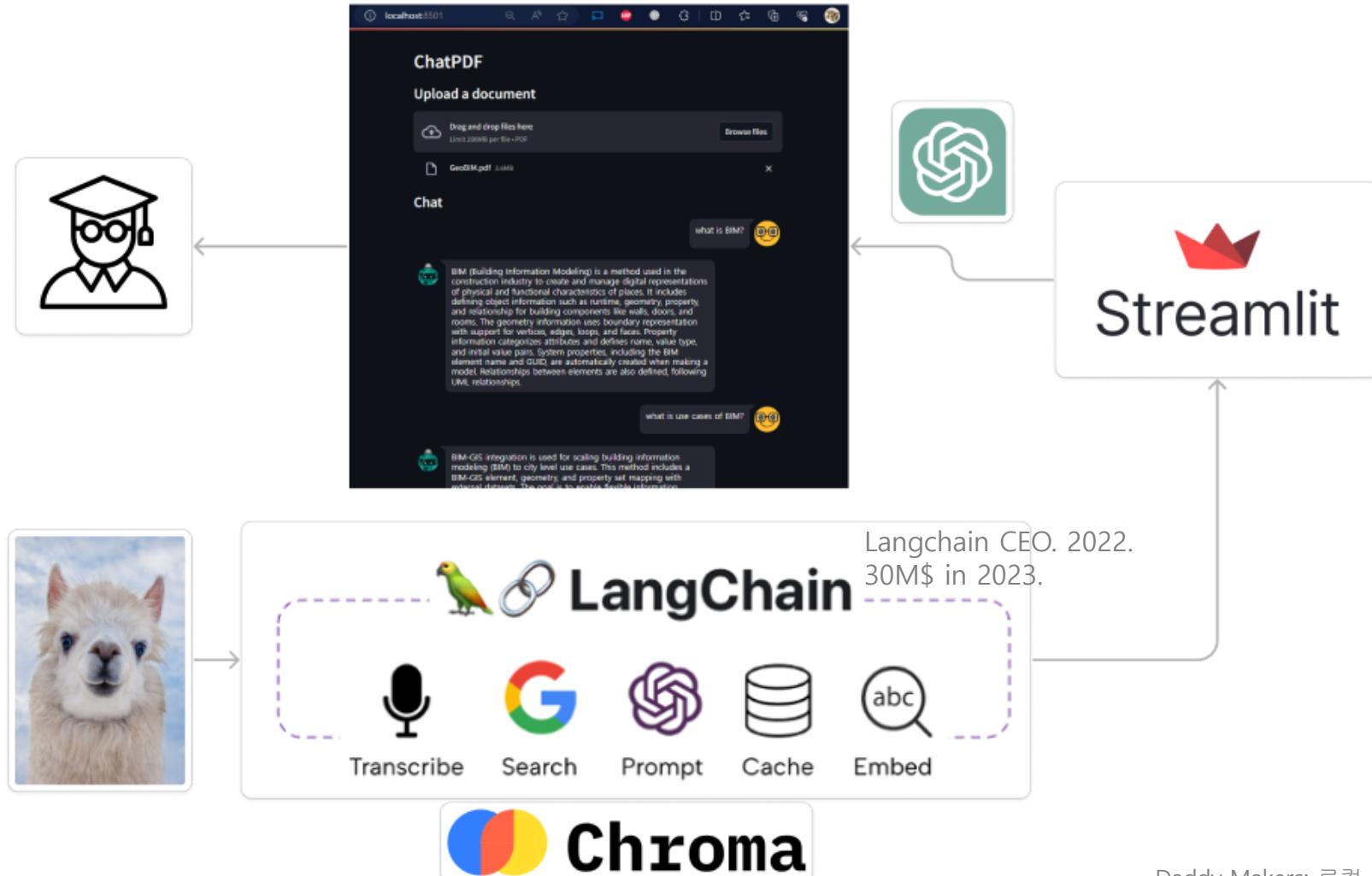
다음 명령을 터미널 명령창에서 실행

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```



# LLM 과 AI Agent Ollama 와 오픈소스모델

## 예시

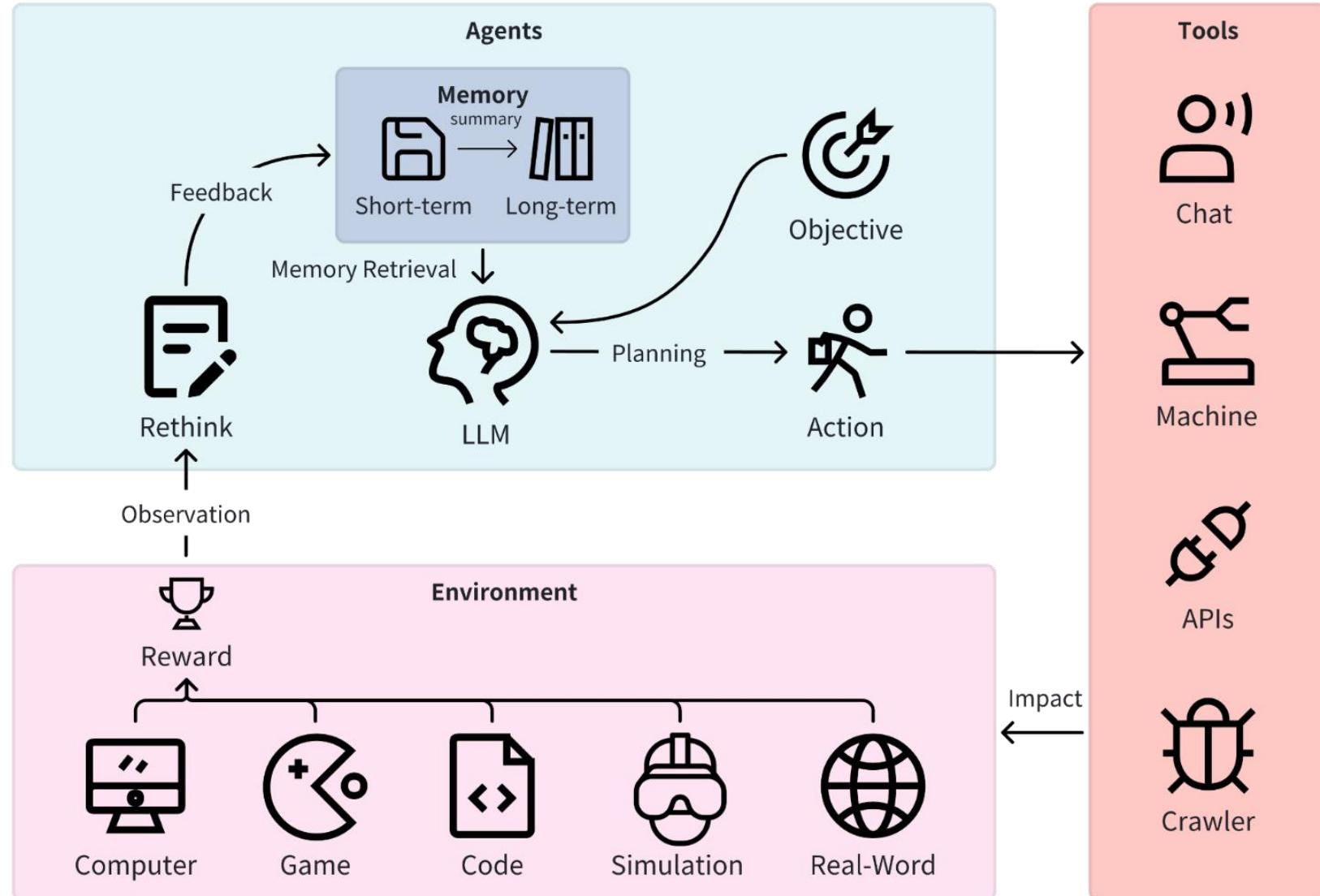


Daddy Makers: 로컬 호스트  
LLM 오픈소스 OLLAMA 기반  
PDF 지식 챗봇 서비스 간단히  
만들어보기

# LLM 과 AI Agent

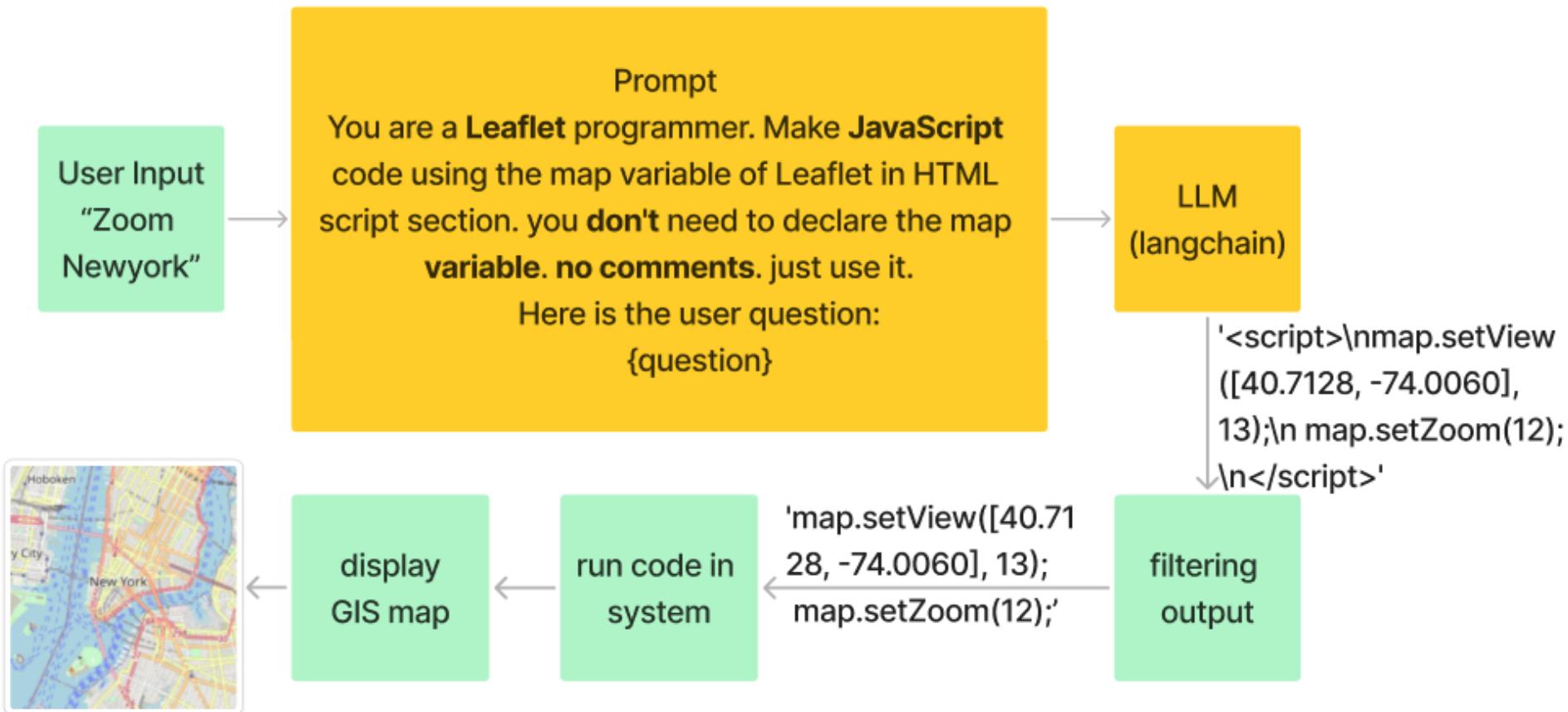
AI Agent

## 멀티 에이전트 모델



# LLM 과 AI Agent

예시



# LLM 과 AI Agent

```
local_llm = 'llama3'
llm = ChatOllama(model=local_llm, temperature=0) # format="json",

def text_to_geo_code(query_text):
    prompt = " " + query_text

    prompt = PromptTemplate(
        template="""You are a Leaflet programmer. Make JavaScript code using the map variable
of Leaflet in HTML script section. I need script section. you don't need to declare
the map variable. no comments. just use it.
Here is the user question:
{question}
""",
        input_variables=["question"],
    )

    text_to_geo_chain = prompt | llm | strOutputParser()
    generated_code = text_to_geo_chain.invoke({"question": query_text})

    script_regex = r"<script>(.*)</script>"
    match = re.search(script_regex, generated_code, re.DOTALL)

    code = ''
    if match:
        code = match.group(1).strip() # Return the extracted code

return code
```

# LLM 과 AI Agent AI Agent

## 예시

Screenshot of a web-based infrastructure management system (Infra Connect) running on localhost:8000.

The dashboard shows the following metrics:

- ISSUE**: 89 (View Details)
- OPEN ISSUE**: 11 (View Details)
- PROGRESS**: 152 (View Details)
- CLOSE ISSUE**: 7 (View Details)

The interface includes:

- Infra map**: A map of Seoul, South Korea, showing various locations marked with red pins. Labels on the map include: 215 m, 342 m, 308 m, 339 m, 124 m, 134 m, and several mountain names like Namsan, Inwangsan, and Gwanaksan.
- IoT Chart**: A chart titled "IoT Data Visualization" showing multiple data series over time. The Y-axis is labeled "IT datasets" with values \$200 and \$300. The X-axis shows time intervals. A dropdown menu indicates the data source is "Accelerometer".

# LLM 과 AI Agent AI Agent Design Pattern

## Tools and function call

The screenshot shows the Hugging Face Datasets interface for a dataset named "xlam-function-calling-60k-parsed". The dataset has one split, "train", containing 60k rows. The interface displays two columns of data: "messages" and "tools".

The "messages" column contains JSON objects representing user interactions. One entry is:

```
[{"content": "Determine the peak of the sequence [44, 46, 48, 50, 52] and the light travel time for 25 light years.", "role": "user", ...}, {"content": "Is 'https://www.apple.com' available in the Wayback Machine on September 9, 2015?", "role": "user", ...}, {"content": null, "role": "assistant", "tool_calls": [{"function": {"arguments": "{\"url\": \"https://www.apple.com\", \"timestamp\": \"20150909\"}", "name": "availability"}]}, {"content": "Can you fetch poor quality backlinks for the domain 'example.com'? Also, generate a QR code for the text 'Visit our...'", "role": "user", ...}, {"content": "Fetch auto-complete results for 'winter coat' from Kohls and 'kitchen appliance' from Wayfair.", "role": "user", ...}], [{"type": "function", "parameters": {"name": "find_peak_element", "type": "object", "properties": {"nums": {"type": "array", "description": "The array of numbers to find the peak element from."}}}, {"type": "function", "parameters": {"name": "availability", "type": "object", "properties": {"url": {"type": "string", "description": "The URL to check for availability in the Wayback Machine."}, "timestamp": {"type": "string", "description": "The timestamp to look up in Wayback. If not specified, the most recent available capture is returned. The format of the timestamp is 1-14 digits (YYYYMMDDhhmmss). Defaults to '20090101'.", "default": "20090101"}, "callback": {"type": "string", "description": "An optional callback to produce a JSONP response. Defaults to None."}}}, {"type": "function", "parameters": {"name": "auto_complet", "type": "object", "properties": {"query": {"type": "string", "description": "The search query for auto-completion."}}}], [{"type": "extra", "text": "The SQL console is powered by DuckDB WASM and runs entirely in the browser. Get started by typing a query or selecting a view from the sidebar."}]]
```

The "tools" column contains JSON objects representing AI functions. One entry is:

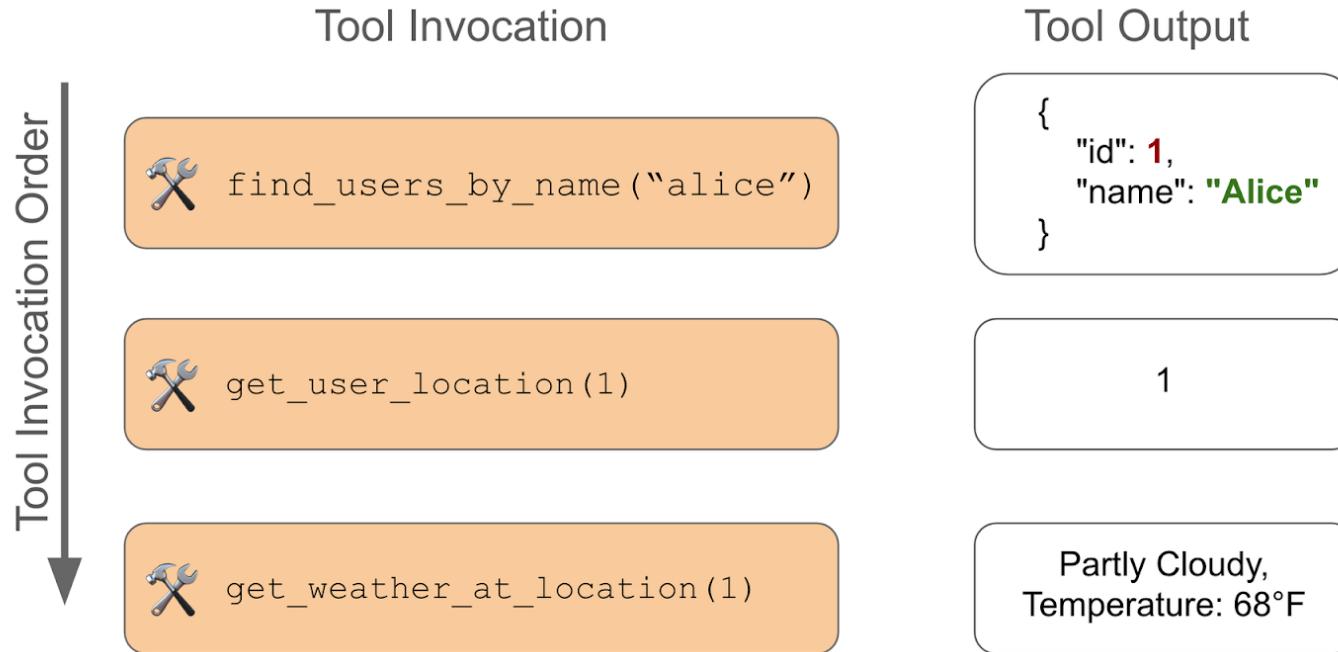
```
{"type": "function", "parameters": {"name": "find_peak_element", "type": "object", "properties": {"nums": {"type": "array", "description": "The array of numbers to find the peak element from."}}}, {"type": "function", "parameters": {"name": "availability", "type": "object", "properties": {"url": {"type": "string", "description": "The URL to check for availability in the Wayback Machine."}, "timestamp": {"type": "string", "description": "The timestamp to look up in Wayback. If not specified, the most recent available capture is returned. The format of the timestamp is 1-14 digits (YYYYMMDDhhmmss). Defaults to '20090101'.", "default": "20090101"}, "callback": {"type": "string", "description": "An optional callback to produce a JSONP response. Defaults to None."}}}, {"type": "function", "parameters": {"name": "auto_complet", "type": "object", "properties": {"query": {"type": "string", "description": "The search query for auto-completion."}}}], [{"type": "extra", "text": "The SQL console is powered by DuckDB WASM and runs entirely in the browser. Get started by typing a query or selecting a view from the sidebar."}]
```

The interface also includes an "Ask AI to help write your query..." input field and a sidebar with various tabs like "Dataset card", "Data Studio", "Files and versions", and "Community".

# LLM 과 AI Agent AI Agent Design Pattern

## Tool과 Function Call

? Is it likely that Alice needs an umbrella now?



! Based on the current weather, it is partly cloudy with a temperature of 68°F at Alice's location. Therefore, it is unlikely that Alice needs an umbrella right now.

# LLM 과 AI Agent AI Agent Design Pattern

## Tool과 Function Call

```
load_dotenv()
hf_token = os.getenv("HF_API_KEY")
login(token=hf_token)

model_name = "google/gemma-3-4b-it"
dataset_name = "Salesforce/xlam-function-calling-60k"

peft_config = LoraConfig(
    lora_alpha=16,
    lora_dropout=0.05,
    r=16,
    task_type="CAUSAL_LM", ...
)

trainer = SFTTrainer(
    model=model,
    args=training_arguments,
    train_dataset=dataset["train"],
    eval_dataset=dataset["test"],
    # tokenizer=tokenizer,
    peft_config=peft_config,
)

trainer.train()
trainer.save_model()
```

# LLM 과 AI Agent AI Agent Design Pattern

## Tool과 Function Call

```
User query:  
What are the roots of the quadratic equation 2x^2 - 3x + 1 = 0?<end_of_turn><eos>  
<start_of_turn>assistant  
<function_call>  
{"name": "solve_quadratic", "arguments": {"a": 2, "b": -3, "c": 1}}  
</function_call><end_of_turn><eos>  
<start_of_turn>model  
  
Converting train dataset to ChatML: 100%|██████████| 54000/54000 [00:04<00:00, 11761.83 examples/s]  
Adding EOS to train dataset: 100%|██████████| 54000/54000 [00:03<00:00, 17786.93 examples/s]  
Tokenizing train dataset: 100%|██████████| 54000/54000 [00:20<00:00, 2575.83 examples/s]  
Truncating train dataset: 100%|██████████| 54000/54000 [00:00<00:00, 462257.72 examples/s]  
Converting eval dataset to ChatML: 100%|██████████| 6000/6000 [00:00<00:00, 11820.43 examples/s]  
Adding EOS to eval dataset: 100%|██████████| 6000/6000 [00:00<00:00, 18528.33 examples/s]  
Tokenizing eval dataset: 100%|██████████| 6000/6000 [00:02<00:00, 2546.05 examples/s]  
Truncating eval dataset: 100%|██████████| 6000/6000 [00:00<00:00, 317345.61 examples/s]  
No label_names provided for model class `PeftModelForCausalLM`. Since `PeftModel` hides base models automatically within `Trainer`. Note that empty label_names list will be used instead.  
wandb: WARNING The `run_name` is currently set to the same value as `TrainingArguments.output_dir`. If the `TrainingArguments.run_name` parameter.  
wandb: Using wandb-core as the SDK backend. Please refer to https://wandb.me/wandb-core for more information.  
wandb: Currently logged in as: mac999 to https://api.wandb.ai. Use `wandb login --relogin` to force relogin.  
wandb: Tracking run with wandb version 0.19.9  
wandb: Run data is saved locally in F:\projects\func-call-gemma3-finetune\wandb\run-20250609_101702-jmwjveij  
wandb: Run `wandb offline` to turn off syncing.  
wandb: Syncing run gemma-3-4b-it-thinking-function_calling-v0  
wandb: View project at https://wandb.ai/mac999/huggingface  
wandb: View run at https://wandb.ai/mac999/huggingface/runs/jmwjveij | 0/5061 [00:00<?, ?it/s]`use_cache=False`.
```

# LLM 과 AI Agent AI Agent Design Pattern

## Function Call 주의사항

Daddy Makers: Gemma3 기반 Ollama 활용 AI 에이전트 개발 핵심 Function Call 구현해보기

### 프롬프트 설계의 명확성

함수 호출이 필요한 상황, 호출 방식(JSON 포맷 등), 호출 예시를 SYSTEM\_MESSAGE에 명확하게 안내  
"질문에 답변하기 위해 함수 호출이 필요하다고 판단되면 반드시 아래 JSON 형식으로만 응답하라. 다른 텍스트나 설명은 절대 포함하지 마라."

### 함수 정의의 구체성

함수의 목적, 파라미터, 반환값, 사용 예시를 상세하게 기술. 각 파라미터의 타입, 필수 여부, 설명을 명확히 함

### 예시 기반 Few-shot Prompting

SYSTEM\_MESSAGE 또는 user message에 함수 호출이 필요한 질문과 그에 대한 올바른 함수 호출 예시를 여러 개 포함

### 함수 호출 실패 시 재시도 로직

모델이 함수 호출을 하지 않거나 잘못된 형식으로 응답하면, 내부적으로 "함수 호출이 필요합니다. 반드시 JSON 형식으로만 응답하세요."와 같은 추가 프롬프트로 재요청

### 출력 파싱의 견고성

모델이 JSON 외의 텍스트를 섞어서 반환 가능. 파싱 로직에서 JSON 부분만 추출하거나, 불완전한 JSON도 최대한 보완해서 파싱

### 함수 호출 의도 강화 프롬프트

SYSTEM\_MESSAGE에 "함수 호출이 필요한 상황에서는 반드시 함수 호출을 우선적으로 고려하라"는 문구를 추가. "만약 함수 호출이 필요하지 않다고 판단되면, 그 이유를 설명하지 말고 바로 답변만 하라." 등 불필요한 설명을 억제

### 모델 버전 및 파라미터 최적화

함수 호출에 최적화된 모델을 사용. temperature, top\_p 등 파라미터를 낮춰 일관된 응답을 유도

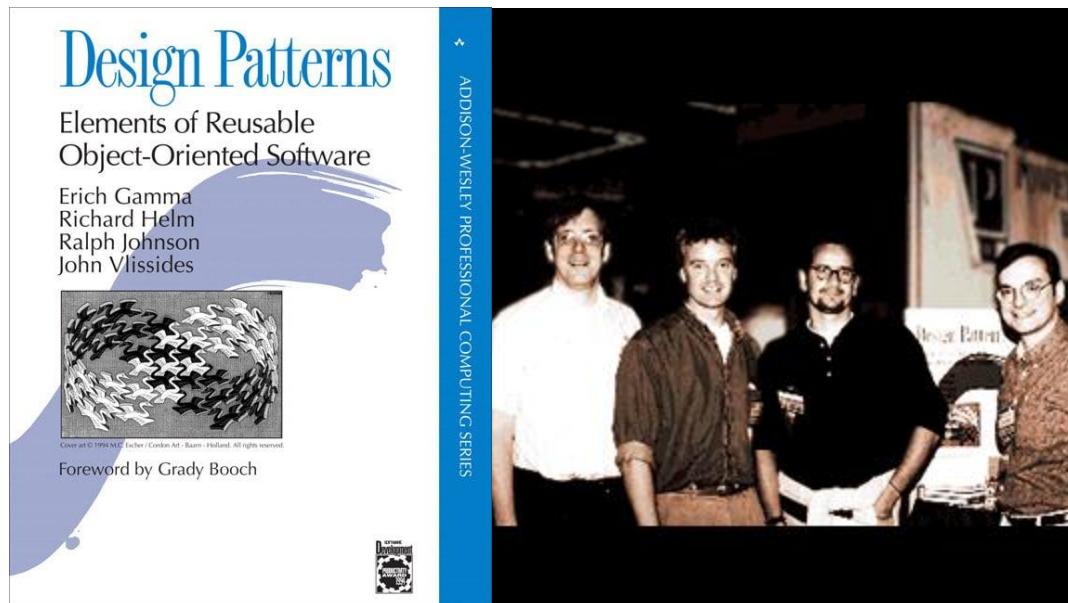
### 함수 호출 실패 케이스 수집 및 개선

실제 사용자 입력 중 함수 호출이 누락된 사례를 수집하여, SYSTEM\_MESSAGE나 예시 프롬프트를 지속적으로 개선

# LLM 과 AI Agent

## AI Agent Design Pattern

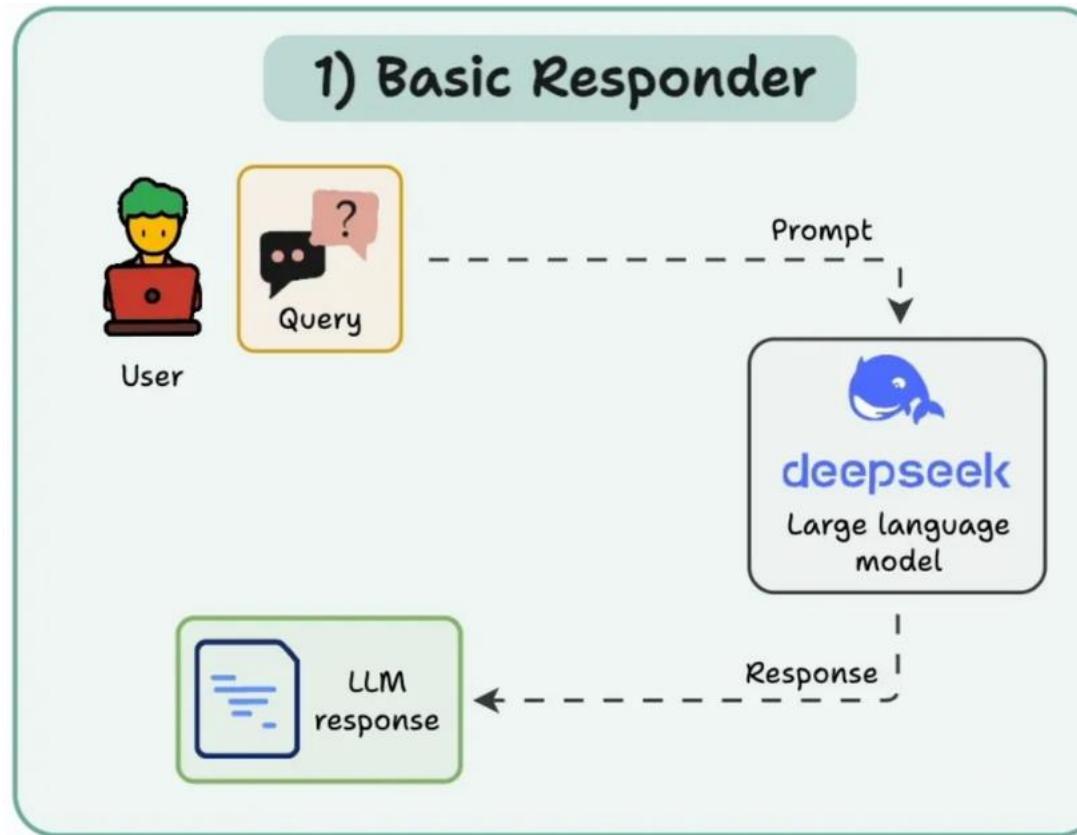
디자인 패턴의 개념은 1994년 "Gang of Four (GoF)"라고 불리는 에리히 감마(Erich Gamma), 리처드 헬름(Richard Helm), 랄프 존슨(Ralph Johnson), 존 블리시디스(John Vlissides) 네 명의 저자가 『Design Patterns: Elements of Reusable Object-Oriented Software』에서 처음 정립한 것으로, 객체지향 소프트웨어 설계에서 반복적으로 등장하는 문제에 대한 해법을 패턴화한 것이 그 출발점



# LLM 과 AI Agent

AI Agent Design Pattern

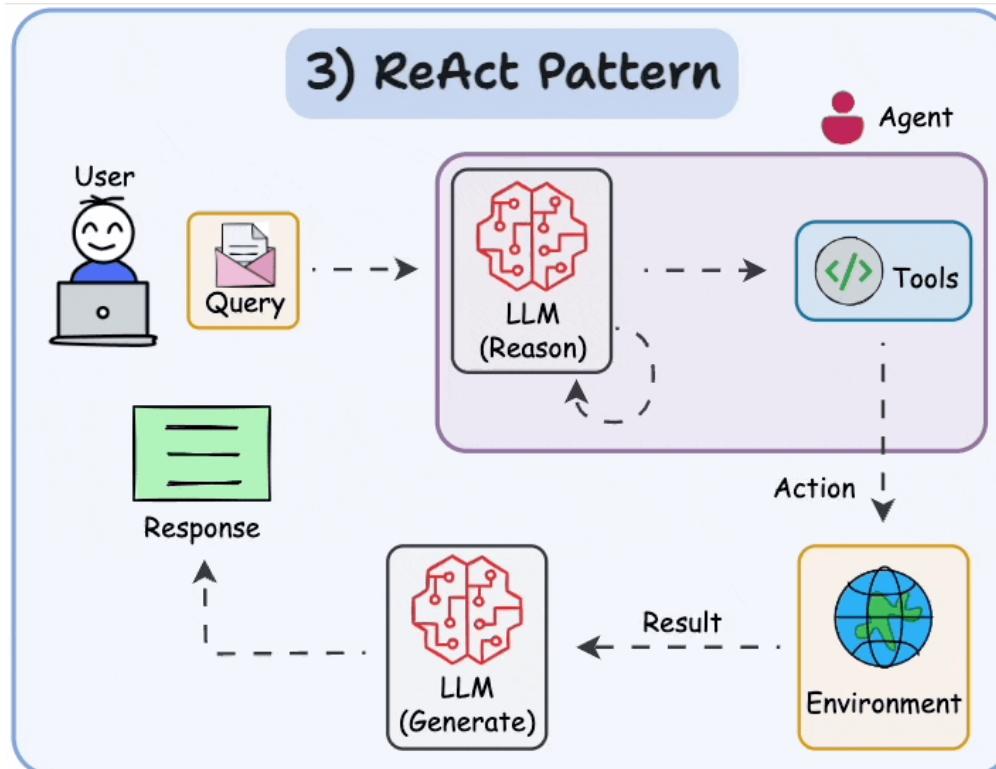
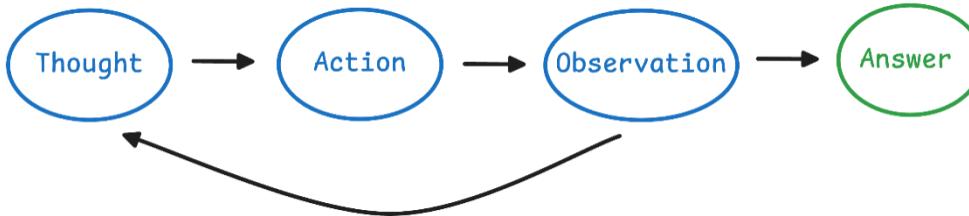
## Basic Responder



# LLM 과 AI Agent

## AI Agent Design Pattern

### ReAct



Question: What are the 3 Newton's laws on motion?

Thought: I should look at the database too see what I can query.

Action: rag\_search

Action Input: <input to search the vector database using rag\_search>: "Newton's three laws on motion"

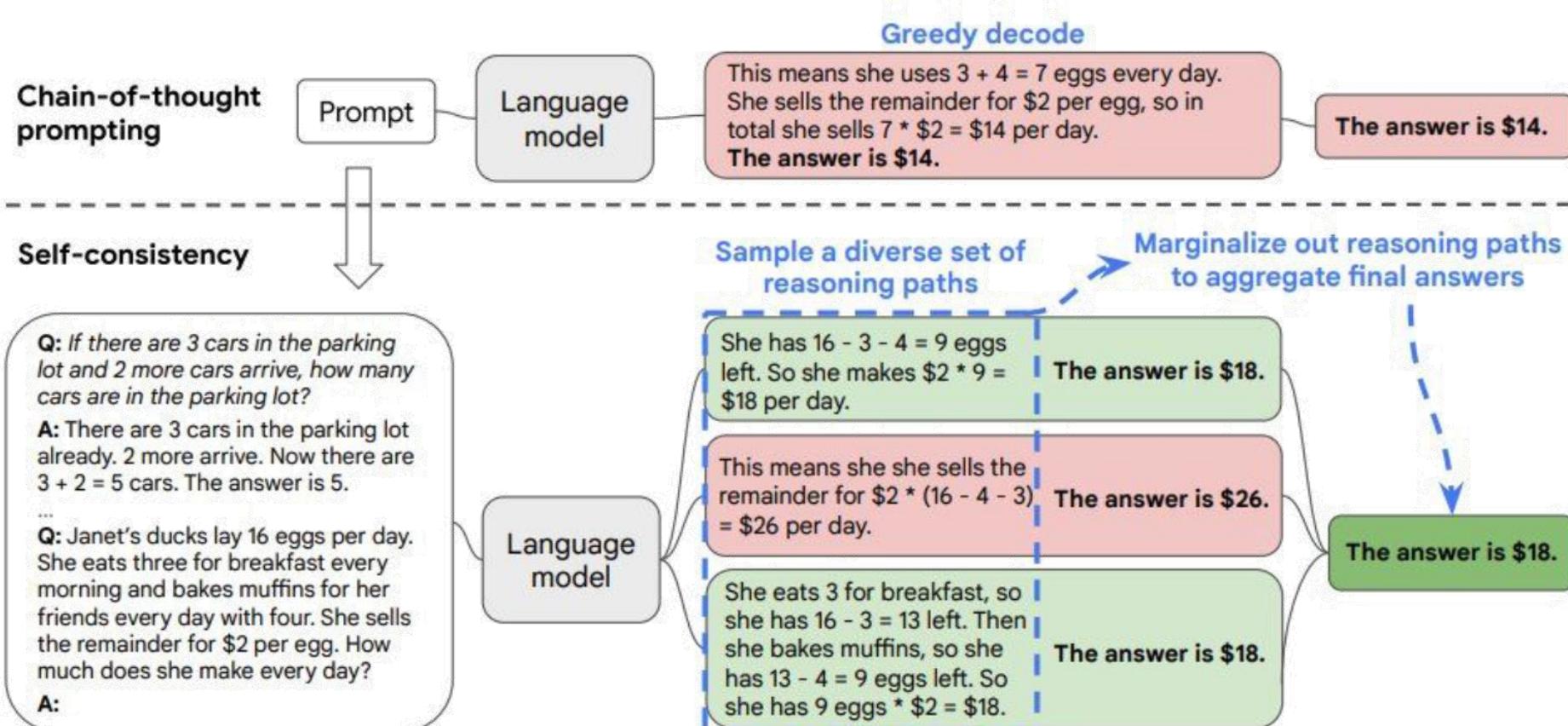
Observation: Newton's three laws of motion applied to classical mechanics. They state....

Thought: I have the final answer. No further tool usage required.

Final Answer: Newton's three laws of motion applied to classical mechanics. They state....

# LLM 과 AI Agent

## AI Agent Design Pattern



# LLM 과 AI Agent AI Agent Design Pattern

## search\_web

**Question:** What are the lens equations?

**Thought:** I should look at the database too see what I can query.

**Action:** `rag_search`

**Action Input:** <input to search the vector database using `rag_search`>: "What are the lens equations?"

**Observation:** No relevant results.

**Thought:** I should use `search_web`

**Action:** `search_web`

**Action Input:** <input to search the web `search_web`>: "What are the lens equations?"

**Observation:** The lens equation expresses the quantitative relationship between the object distance....,

<https://www.physicsclassroom.com/classes/refrn/Lesson-5/The-Mathematics-of-Lenses>

**Thought:** I have the final answer. No further tool usage required.

**Final Answer:** According to the website...

## remember/checkpoint

**Question:** Checkpoint

**Thought:** I should checkpoint the current progress.

**Action:** `checkpoint`

**Action Input:** <input to checkpoint `checkpoint`>: current conversation, current topic in progress, function call stack etc.

**Observation:** The current conversation state, and the current tool call stack has been saved to long term memory.

**Thought:** I have the final answer. No further tool usage required.

**Final Answer:** Checkpoint successful

## pop\_quiz

**Question:** Pop Quiz

**Thought:** I should generate a pop quiz using the template on the current topic of study

**Action:** `pop_quiz`

**Action Input:** <input to pop quiz `pop_quiz`>: current conversation, current topic in progress, previous topics completed

**Observation:** Pop quiz generated with topic A, B etc. with display UI and API backend.

**Thought:** I have the final answer. No further tool usage required.

**Final Answer:** Here is your pop quiz, please enter the answers for the respective questions and press submit at the end.

this is interesting since `pop_quiz` needs to interact with another API/system to generate the question answer UI and do something on submission

# LLM 과 AI Agent AI Agent Design Pattern

## 싱글 스킬 패턴 (Single Skill Agent)

이 패턴은 하나의 태스크에 특화된 에이전트를 구축할 때 사용. 예컨대, 사내 문서 기반 FAQ 검색, 날씨 응답, 이메일 요약 등 단일 기능에 초점을 맞추어, 파인튜닝된 LLM 혹은 RAG 기반 구조로 구성. 입력은 간단한 질의이며, 임베딩 검색기(retriever)와 하나의 LLM으로 구성. 단일 기능을 빠르게 서비스화하기에 적합

## 멀티 스킬 라우팅 패턴 (Multi-Tool Routing Agent)

이 패턴은 입력에 따라 다양한 도구(tool) 또는 스킬을 선택적으로 실행하는 구조. 예를 들어 "계산해줘"라는 명령은 파이썬 실행 도구를, "날씨 알려줘"는 외부 API 도구를, "문서 요약해줘"는 LLM을 호출하는 식. MCP 기반 도구 시스템, OpenAI Function/Tool Calling, LangChain ReAct 등에서 구현 가능, 클라이언트는 입력을 분류하는 선택기(Classifier 또는 Planner)를 내장.

## 전문가 집단 패턴 (Expert Ensemble Agent)

이 패턴은 서로 다른 도메인에 특화된 LLM 또는 에이전트를 조합해 협력적인 응답을 생성. 예를 들어, 의학 질문은 Med-PaLM, 법률 질문은 GPT-Legal로 라우팅하거나, 모든 전문가의 답변을 받아 다수결 또는 점수 기반으로 결합하는 방식. 다중 에이전트를 조합하는 고급 전략으로, 전문성 기반 서비스에 적합

## 멀티모달 에이전트 패턴 (Multimodal Retrieval Agent)

텍스트 외에도 이미지, 음성, 동영상, PDF, PPT, 음악 등 다양한 포맷의 데이터를 동시에 검색하고 분석하는 구조. 이를 위해 멀티모달 임베딩 모델(CLIP, Whisper, MusicLM 등)이 사용되며, 통합된 벡터 검색 인덱스에서 유사도를 기준으로 문서를 탐색

# LLM 과 AI Agent AI Agent Design Pattern

## 메타 에이전트 패턴 (Meta Agent)

에이전트가 다른 에이전트들을 통제하거나 조합할 수 있는 구조. 사용자의 복합 지시문("문서를 요약하고 PPT 만들어줘")을 받아, 먼저 요약 도구를 호출하고, 결과를 기반으로 생성 도구를 다시 호출하는 일련의 워크플로우를 구성. Self-RAG, Planner + Tool Executor, ReAct 기반 구조 등에서 이 패턴을 구현할 수 있음

## 문맥 보강 패턴 (Context Enrichment Agent)

이 패턴은 LLM의 입력 컨텍스트를 외부 실시간 정보(API, DB, 클라우드 데이터 등)로 보강하는 방식. 주로 MCP(Model Context Protocol) 구조를 기반으로 하며, 서버가 제공하는 다양한 도구에서 실시간 정보를 얻어, 이를 LLM 입력에 삽입. 예를 들어, 환율 API를 호출한 후 결과를 사용자 질의와 함께 LLM에 전달하는 방식

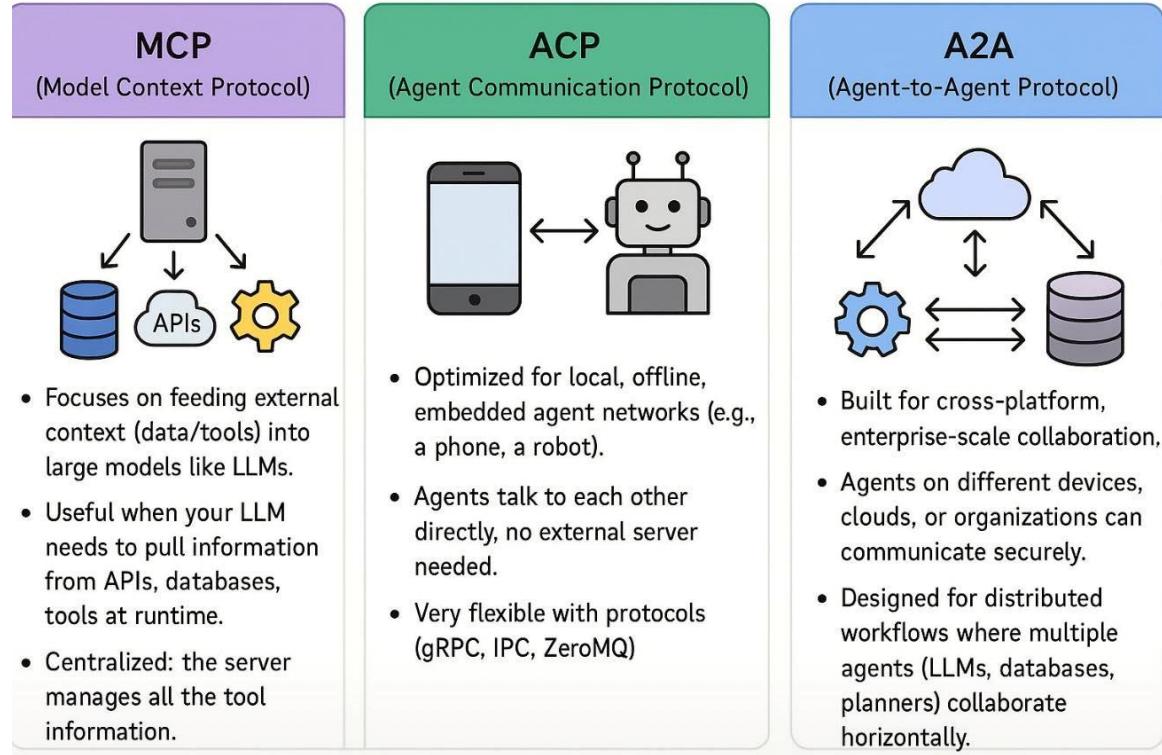
# LLM 과 AI Agent MCP

Model Context Protocol(MCP)은 2023년 중반, 클로드(Claude) 개발사로 알려진 Anthropic에서 제안

MCP는 LLM(Large Language Model) 중심의 아키텍처에서 외부 데이터 소스 또는 도구를 연결하는 데 최적화된 통신 프레임워크로, 주로 툴 호출 및 컨텍스트 주입을 위해 설계

Agent Communication Protocol(ACP)은 Anthropic 및 일부 커뮤니티 기반 프로젝트에서 MCP 이후의 보완 프로토콜로 제안된 것으로 알려져 있으며, 특히 로컬 에이전트 네트워크에서의 효율적인 통신을 목적. ACP는 MCP와 달리 중앙 서버가 필요 없는 점에서 구조적 차이

Agent-to-Agent Protocol(A2A)은 2024년부터 Microsoft, Hugging Face, OpenAI 등 복수의 기업 및 오픈소스 커뮤니티에서 논의되기 시작한 개방형 표준. A2A는 대규모 분산 환경에서 복수의 에이전트가 서로 협력할 수 있는 수평적 통신 프레임워크를 지향



# LLM 과 AI Agent MCP

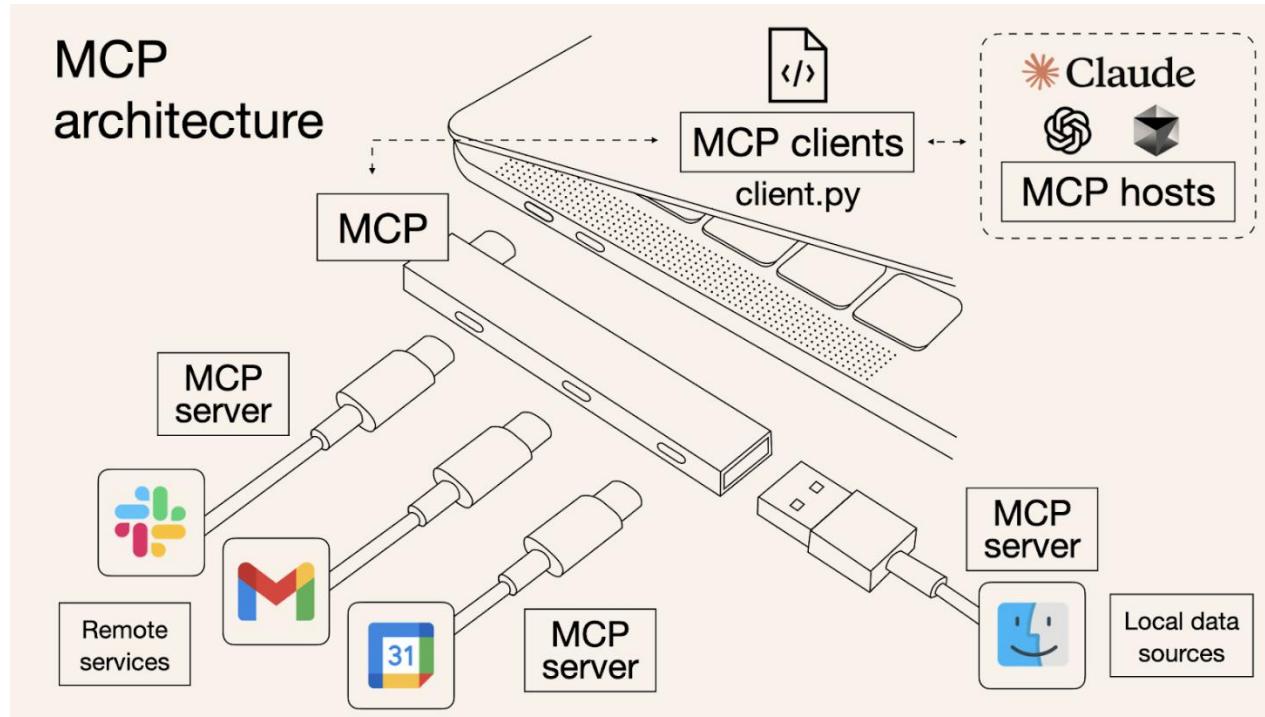
MCP는 애플리케이션이 LLM에 컨텍스트를 제공하는 방식을 표준화한 개방형 프로토콜. USB-C 포트처럼, 다양한 도구와 데이터를 LLM에 연결하는 통합된 인터페이스 역할

클로드 데스크탑: [Download - Claude](#). 호스트 역할. 현재, 클로드, Github Copilot, Cursor, OpenAI 등 지원

Node.js — Run JavaScript Everywhere. NODE.JS 기반 서버 도구를 실행하는 역할

파이썬: 안정 버전을 설치함(예를 들어, 현재 시점 3.11 이상, 3.12 이하 버전 권장)

최신 파이썬 패키지 설치 UV 도구: [UV](#). extremely fast Python package and project manager, written in Rust.



# LLM 과 AI Agent MCP

Claude — Control+Alt+Space

Exploring Minecraft Coder Pack (MCP) Tools

YE 난 품이야. 개발자야. 요즘 시티팝에 대해 흥미가 있어. 유명한 시티팝 가수를 알려줘. 그 정보를 리스트 해서 REPORT.TXT 란 이름으로 내 폴더에 저장해줘.

안녕하세요, 품님! 시티팝에 관심이 있으시군요. 유명한 시티팝 가수들에 대한 정보를 모아서 REPORT.TXT 파일로 저장해 드리겠습니다.

write\_file

죄송합니다. 파일 저장 시 접근 권한 문제가 발생했네요. 현재 허용된 디렉토리에만 파일을 저장할 수 있습니다. 허용된 디렉토리를 확인해 보겠습니다.

list\_allowed\_directories

확인된 허용 디렉토리에 파일을 저장해 보겠습니다:

write\_file

품님, 시티팝 가수들에 대한 정보를 성공적으로 C:\Users\MAC\Desktop\REPORT.TXT 파일에 저장했습니다.

파일에는 다음 내용이 포함되어 있습니다.

- 일본의 대표적인 시티팝 가수 10명
- 한국의 시티팝 계열 아티스트 5명
- 현대 해외 시티팝 영향 아티스트 4명
- 시티팝의 주요 특징
- 추천 시티팝 앨범 10개

시티팝에 더 관심이 있으시다면, 특정 아티스트나 앨범에 대해 더 자세한 정보가 필요하신지요?

Claude에게 답변하기

Claude 3.7 Sonnet

Line 1, Column 1

File Edit Selection Find View Goto Tools Project Preferences Help

OPEN api-key.txt REPORT.TXT daude\_desktop\_config.json

FOLD

## 유명한 시티팝 가수

1. 타쓰로 야마시타(Tatsuro Yamashita) - '크리스마스 이브', 'Ride on Time' 등의 히트곡으로 알려진 시티팝의 전설적인 아티스트

2. 마리야 타케우치(Mariya Takeuchi) - 'Plastic Love'로 전 세계적으로 유명해진 시티팝의 여왕

3. 타에코 오니uki(Taeiko Onuki) - '4:00 AM', 'Mignonne' 등 섬세한 음악성으로 유명

4. 어누아누(Anri) - 'Last Summer Whisper', 'Shyness Boy' 등의 히트곡으로 사랑받는 가수

5. 메구미 야마자키(Megumi Yamazaki) - '미요코하마 타소가레(Yokohama Twilight)' 등으로 유명

6. 히데아키 토리우치(Hideaki Tokunaga) - 감미로운 목소리로 많은 히트곡을 낸 남성 시티팝 가수

7. 하로시 사토(Hiroshi Sato) - 'Awakening'과 같은 명반으로 알려진 키보드 연주자이자 프로듀서

8. 후쿠이 유키히로(Yukihiro Fukui) - Cassiopeia를 비롯한 다양한 밴드에서 활동한 건반 연주자

9. 마코토 마쓰시타(Makoto Matsushita) - 'First Light'와 같은 명반으로 알려진 기타리스트

10. 카도마츠 토시키(Toshiki Kadomatsu) - 'After 5 Clash'로 알려진 시티팝의 선구자

## 한국 시티팝 계열 아티스트

1. 유카리(Yukari) - 80년대 시티팝 감성을 재현하는 현대 한국 아티스트

2. 선우정아 - 레트로한 감성의 음악을 선보이는 싱어송라이터

3. 윤종신 - 음학적이고 복고적인 음악을 선보이기도 하는 베테랑 가수

4. 박혜진(Park Hye Jin) - 일렉트로닉과 점목된 현대적 시티팝 스타일

5. 뉴진스(NewJeans) - '슈퍼사이', 'ETA' 등 레트로 감성의 팝 음악을 선보이는 아이돌 그룹

## 현대 해외 시티팝 영향 아티스트

1. Yung Bae - 미국의 뮤쳐 펑크, 시티팝 프로듀서

2. Desired - 캐나다의 시티팝, 밴포웨이브 아티스트

3. Macross 82-99 - 액시코 출신의 전자음악 프로듀서

4. Night Tempo - 한국계 미국인 프로듀서, 시티팝 리믹스로 유명

## 시티팝 특징

- 1970년대 후반부터 1980년대 일본에서 유행한 장르

- 서구의 재즈, 펑크, AOR, 소울 음악의 영향을 받음

- 세련된 도시적 감성과 풍요로운 시대상을 반영

- 복잡한 코드 진행과 정교한 편곡이 특징

- 레트로한 신스사이저와 삼스폰, 일렉트릭 피아노 등의 악기 사용

- 최근 인터넷과 유튜브를 통해 전 세계적으로 재조명됨

## 추천 시티팝 앨범

1. Tatsuro Yamashita - "For You" (1982)

2. Mariya Takeuchi - "Variety" (1984)

3. Taeko Onuki - "Sunshower" (1977)

4. Anri - "Timely!!" (1983)

5. Hiroshi Sato - "Awakening" (1982)

6. Toshiki Kadomatsu - "After 5 Clash" (1984)

7. Momoko Kikuchi - "Adventure" (1986)

8. Takako Mamiya - "Love Trip" (1982)

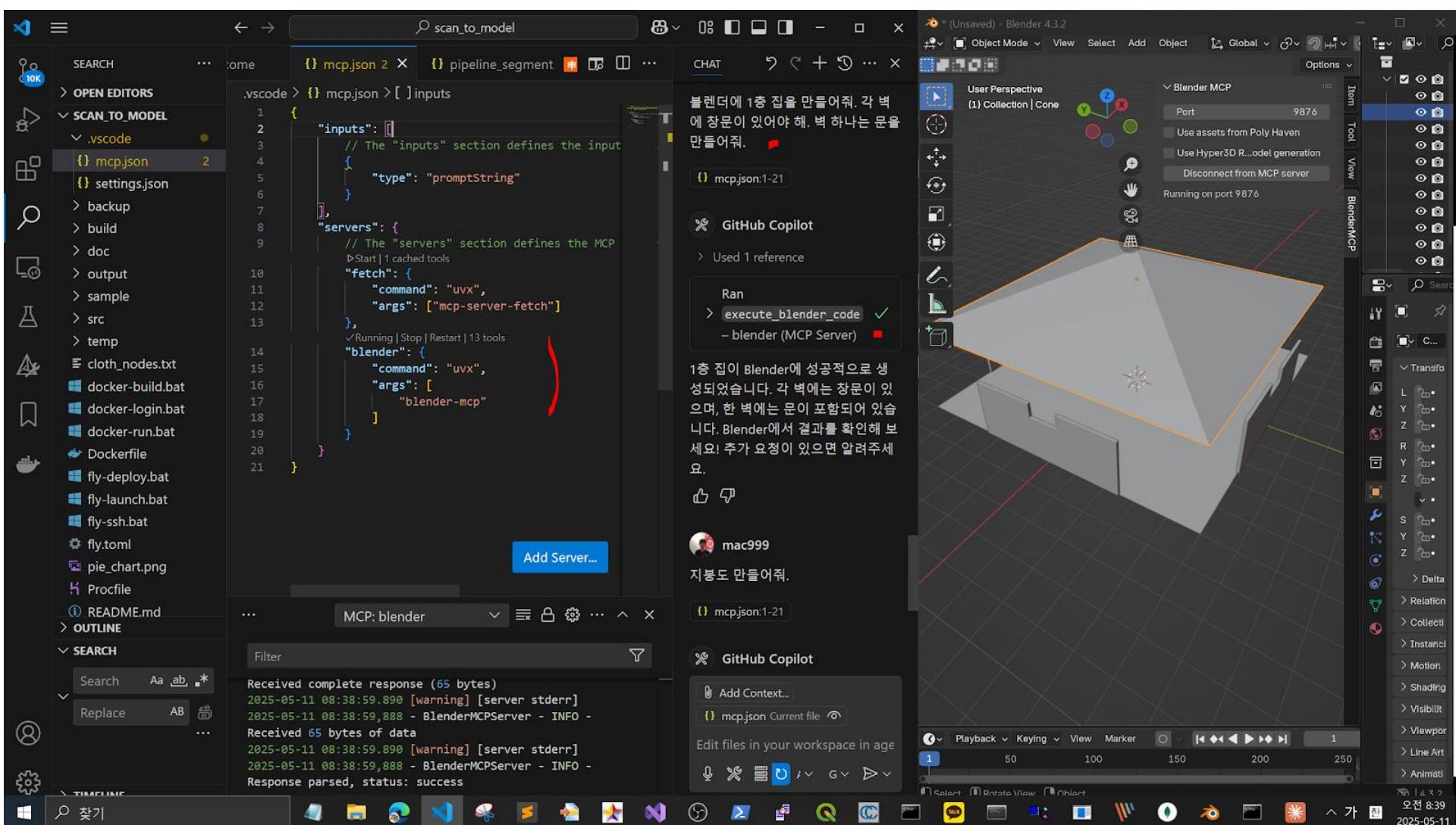
9. Junko Ohashi - "Magical" (1984)

10. Miki Matsubara - "Pocket Park" (1988)

Tab Size: 4 Plain Text

Daddy Makers: 인공지능 AI 에이전트 표준 프로토콜 MCP 개념, 사용, 개발 및 동작 구조 분석하기

# LLM 과 AI Agent MCP



# LLM 과 AI Agent MCP

https://n8n.io

n8n Product Use cases Docs Community Enterprise Pricing GitHub ★ 87,123

**IT Ops can**

- ⚡ On-board new employees and provision accounts

**Sec Ops can**

- ⚡ Enrich security incident tickets

**Dev Ops can**

- ⚡ Convert natural language commands into API calls

**Sales can**

- ⚡ Generate customer insights from grouped reviews

```
graph LR; Start((On 'Create User' form submission)) --> Agent[AI Agent Tools Agent]; Agent --> Manager{Is manager?}; Manager -- true --> AddChannel[Add to channel  
invite: channel]; Manager -- false --> UpdateProfile[Update profile  
updateProfile: user]; Agent --> Anthropic[Anthropic Chat Model]; Agent --> Postgres[Postgres Chat Memory]; Manager --> Microsoft[Microsoft Entra ID  
getAll: user]; Manager --> Jira[Jira Software  
create: user];
```

Powerful Workflow Automation  
Software & Tools - n8n

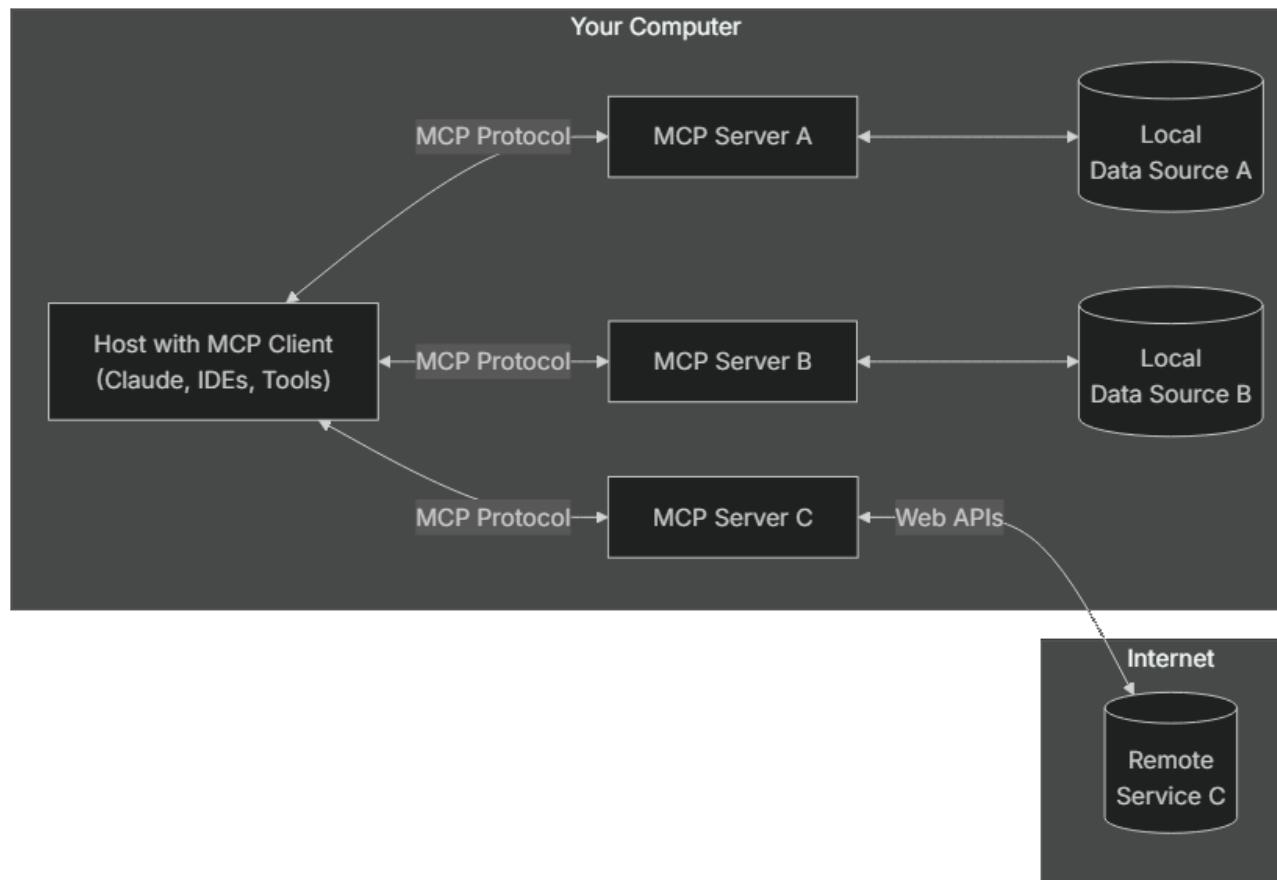
# LLM 과 AI Agent MCP

MCP 호스트: Claude 데스크톱, AI 에이전트, 커서 IDE와 같은 MCP 클라이언트 통합 환경

MCP 서버: 기능을 노출하는 프로그램으로, 도구 등록 및 실행을 처리. Python, Node.js 등 다양한 런타임 환경에서 동작

로컬 데이터 소스: 파일 시스템, DB 등 로컬 자원

원격 서비스: GitHub, Brave Search 등 API 기반 외부 서비스



# LLM 과 AI Agent Gen AI와 Vibe Coding

llasses antighting, persp  
s and su realistic  
eadphon black,  
wi he, pitch  
ish e sene ton  
k gimoiesense lebato  
run5mlr sng, el  
cbsnmrl f ee bzz oo en sap cphy enhpid  
Lny u r li dc tehagap eig r  
hMu knncc oonee se l ad h etgig main  
o ok ioes m foot rg al nl hieheadfirawa  
tc til ooe MM la cacal hiny ucer  
ees,raa0 ccv,, pitnt ik,lcCtyrat  
esolmr15 laslibb th reaun inel C ha  
Pycbmp1 MrUU, i tnemo mdineou,  
yc,bnhs vulge w acil ann Cotr  
re, re, ig, t com  
it, achn vn, i,

Midjurney



Promptlot

## Parent



## Prompt

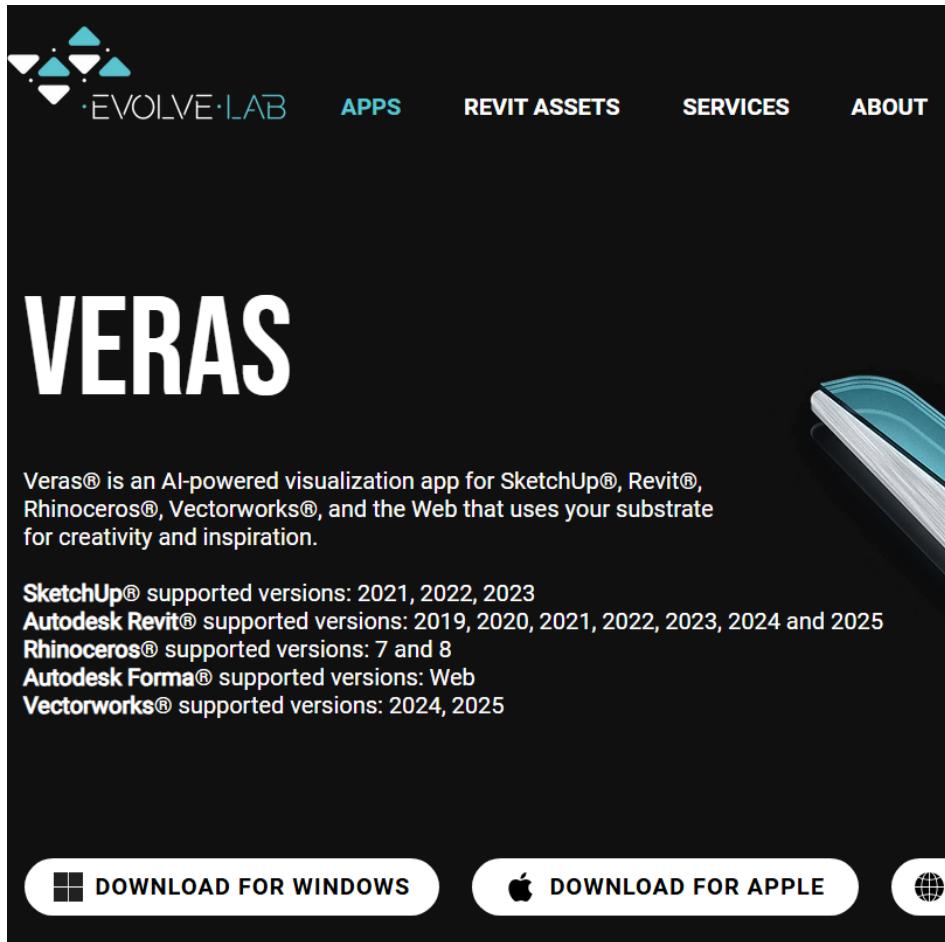
A painting for a house designed by Frank Lloyd Wright and the painting is made by Van Gogh using oil painting rough strokes warm colors dominant color is yellow --ar 16:9 --v 6.1

## Description

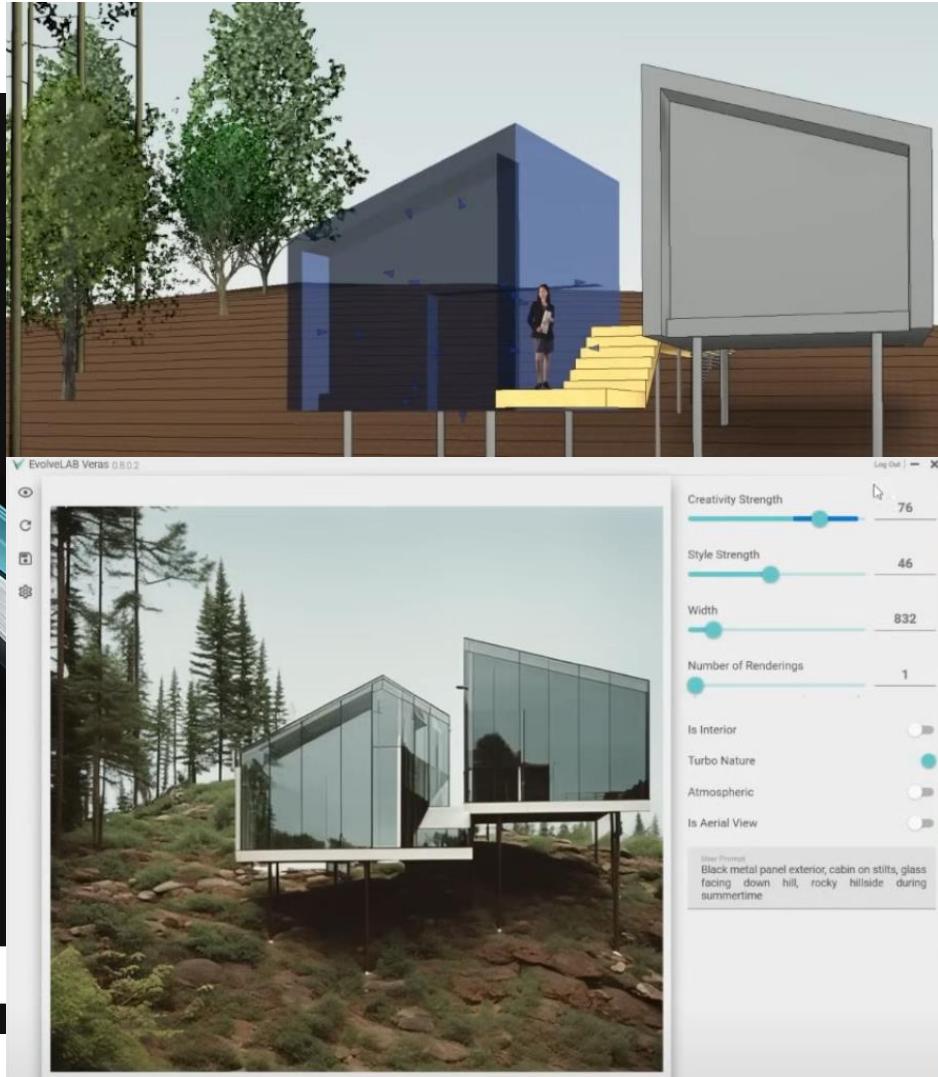
This painting showcases a modernist house with bold yellow autumn trees in the background, emphasizing the blend of architecture and nature.

## Details

# LLM 과 AI Agent Gen AI와 Vibe Coding



The screenshot shows the EvolveLAB Veras website. At the top, there's a navigation bar with a logo of three blue triangles pointing up, followed by the text "EVOLVE·LAB" and menu items "APPS", "REVIT ASSETS", "SERVICES", and "ABOUT". Below this, a large white title "VERAS" is centered on a dark background. Underneath the title, a paragraph describes Veras as an AI-powered visualization app. It lists supported software: SketchUp® 2021, 2022, 2023; Autodesk Revit® 2019, 2020, 2021, 2022, 2023, 2024 and 2025; Rhinoceros® 7 and 8; Autodesk Forma® Web; and Vectorworks® 2024, 2025. At the bottom, there are two download buttons: "DOWNLOAD FOR WINDOWS" with a Microsoft logo and "DOWNLOAD FOR APPLE" with an Apple logo.



VERAS | EvolveLAB

# LLM 과 AI Agent Vibe Coding

[Create](#) [Remix](#) [Edit](#) [3D](#) [Create a new world by describing it. Sketch in guides if you like.](#) [Heart](#) [Download](#) [Share](#)

An open, futuristic interior with organic, flowing architecture. Smooth, curved white walls and ceilings echo the exterior, creating a fluid, airy space. Floor-to-ceiling glass walls allow natural light to flood in, illuminating sleek, minimalist furnishings in white and light gray. Polished marble floors add elegance, while built-in LED lighting highlights architectural curves. Floating sofas, suspended lights, and hidden storage create a clutter-free look, with subtle greenery adding warmth to the modern aesthetic.

522/600

Negative Text  Enhance Prompt

M3 UHD Render

GENERATE



Skybox AI

# LLM 과 AI Agent

Gen AI와 Vibe Coding



Adobe Firefly

# LLM 과 AI Agent Gen AI와 Vibe Coding

## OpenArt

설명: 텍스트로부터 이미지 생성, 스타일 변환, 커스텀 모델 학습 및 사진을 비디오로 변환하는 기능 제공.

링크: <https://openart.ai/>

## Runway ML

설명: 텍스트를 기반으로 비디오 생성 및 편집을 지원하는 AI 콘텐츠 제작 플랫폼.

링크: <https://app.runwayml.com/>

## Sora (OpenAI)

설명: 텍스트 설명을 기반으로 고품질 비디오를 생성하는 OpenAI의 실험적 모델.

링크: <https://sora.chatgpt.com/explore/videos>

## Suno AI (구: Mureka AI)

설명: 텍스트 프롬프트로 음악을 생성하는 생성형 AI 서비스.

링크: <https://www.mureka.ai/>

# LLM 과 AI Agent Gen AI와 Vibe Coding

## D-ID

설명: 사진을 애니메이션화하고, 텍스트나 오디오를 기반으로 말하게 만드는 AI 서비스.

링크: <https://studio.d-id.com/>

## Kling AI

설명: 캐릭터 움직임과 장면 시뮬레이션이 가능한 고성능 비디오 생성 AI.

링크: <https://klingai.com/global/>

## Luma Labs Dream Machine

설명: 간단한 텍스트를 고품질 짧은 동영상으로 변환하는 생성형 AI 모델.

링크: <https://dream-machine.lumalabs.ai/>

## Claude MCP (Model Context Protocol)

설명: Anthropic이 개발한 오픈 프로토콜로, Claude 같은 대형 언어 모델이 외부 도구, 시스템, 데이터 소스와 실시간으로 상호작용할 수 있도록 지원함.

링크: <https://modelcontextprotocol.io/>

# LLM 과 AI Agent Gen AI와 Vibe Coding

바이브 코딩(Vibe Coding)은 특정 분위기나 느낌(vibe)을 조성하거나 반영하는 데 중점을 둔 코딩 방식을 의미

[ChatGPT](#)에 코딩 요청을 해서 생성된 파이썬 같은 코드를 복사&붙여넣기해 프로그램을 완성해 나가는 방법

[Gemini CLI](#), [Claude code CLI](#), [codex CLI](#) 도구를 사용해 프로젝트 파일 및 소스코드를 생성하는 방법

vscode 같은 개발 [IDE](#)와 연동되는 github [copilot](#), [cursor](#), [windsurf](#)와 같은 도구를 사용해 바이브 코딩하는 방법

[Bubble.io](#)나 [Canva](#)와 같은 바이브 코딩 웹서비스에서 직접 요구사항을 입력하여 제공 클라우드에 앱을 생성 빌드 실행하는 방법

# LLM 과 AI Agent Gen AI와 Vibe Coding

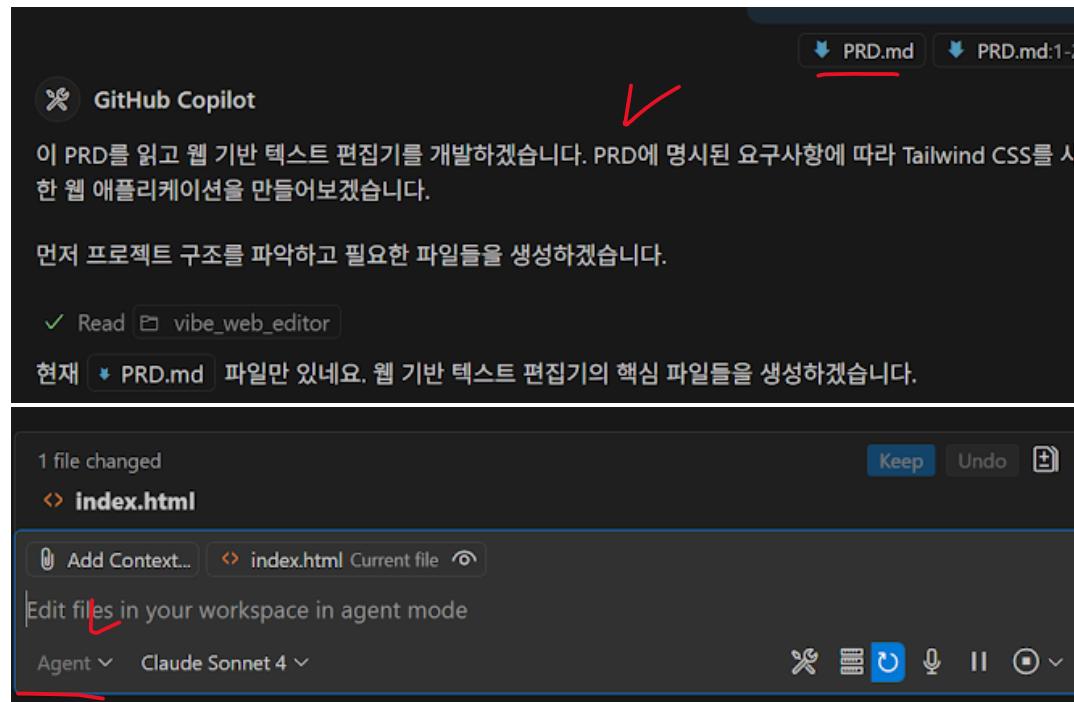
GUI 기반 텍스트 편집기를 개발할꺼야. 파이썬과 Tkinter 라이브러리를 이용할꺼야. 메뉴는 파일 메뉴, 편집 메뉴, 테마 선택 메뉴(다크모드 하위 메뉴 포함), 찾기 메뉴로 구성됨. 이 프로그램 개발을 위한 PRD를 UI 스케치를 포함해 md 포맷으로 작성해.

사용할 라이브러리와 같은 기술 스택을 지정하는 것은 중요

시행착오를 하며 명확한 PRD가 작성될 수 있도록 프롬프트에 정확한 정보를 포함

md 포맷은 좀 더 명확한 문서형식작성을 위해 사용

md포맷은 markdown 파일형식으로 github에서 README 파일 작성할때 표준 형식



# LLM 과 AI Agent Gen AI와 Vibe Coding

# \*\*Python/Tkinter 기반 GUI 텍스트 편집기 제품 요구사항 명세서 (PRD)\*\*

## \*\*5. UI 스케치\*\*

## \*\*1. 개요\*\*

본 문서는 파이썬(Python)과 표준 GUI 라이브러리인 티케이인터(Tkinter)를 사용하여 개발될 디스크톱 텍스트 편집기의 요구사항을 정의합니다. 이 애플리케이션은 기본적인 텍스트 편집 기능과 함께 사용자의 편의를 위한 다크 모드, 찾기 기능 등을 제공하는 것을 목표로 합니다.

## \*\*2. 목표 사용자\*\*

\* \*\*초보 개발자 및 학생\*\*: 간단한 코드나 스크립트, 메모, 과제 등을 작성하고 편집하려는 사용자.

\* \*\*일반 사용자\*\*: Windows 메모장과 같은 가볍고 빠른 텍스트 편집기가 필요한 사용자.

## \*\*3. 핵심 기능\*\*

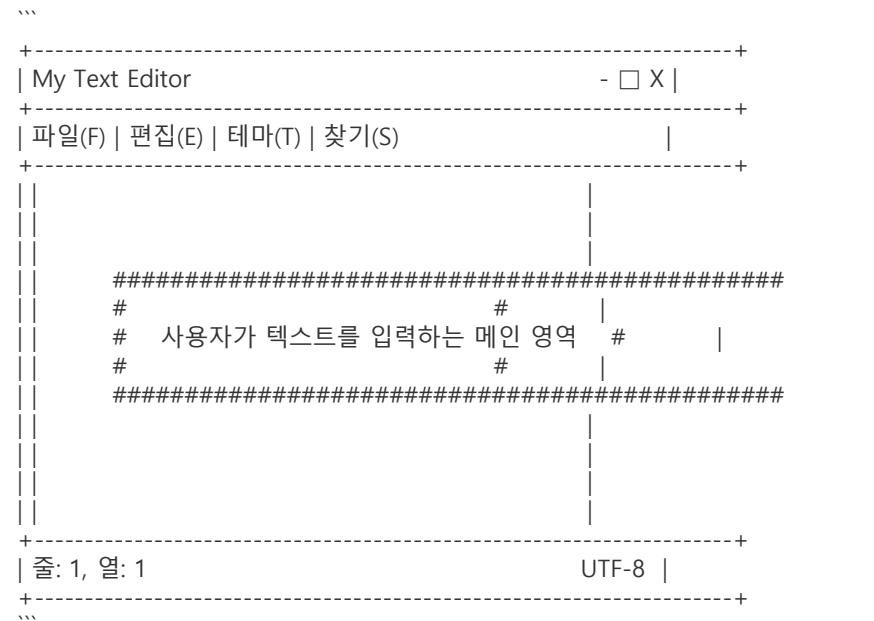
#### \*\*3.1. 파일 메뉴\*\*

기능	단축키	설명
\*\*새 파일\*\*	`Ctrl+N`	새로운 빈 편집 창을 엽니다.
\*\*열기\*\*	`Ctrl+O`	기존 텍스트 파일 (.txt` 등)을 불러옵니다.
\*\*저장\*\*	`Ctrl+S`	현재 파일을 덮어쓰기 저장합니다.
\*\*다른 이름으로 저장\*\*	`Ctrl+Shift+S`	새 이름이나 경로로 파일을 저장합니다.
\*\*종료\*\*	`Alt+F4`	프로그램을 종료합니다. (저장되지 않은 내용이 있으면 경고)

#### \*\*3.2. 편집 메뉴\*\*

기능	단축키	설명
\*\*실행 취소 (Undo)\*\*	`Ctrl+Z`	마지막 작업을 되돌립니다.
\*\*다시 실행 (Redo)\*\*	`Ctrl+Y`	되돌린 작업을 다시 실행합니다.
\*\*잘라내기\*\*	`Ctrl+X`	선택한 텍스트를 잘라내 클립보드에 저장합니다.

애플리케이션의 전체적인 레이아웃 스케치입니다.



\*\*UI 구성 요소:\*\*

\* \*\*제목 표시줄\*\*: 프로그램 이름과 창 조절 버튼이 표시됩니다.

\* \*\*메뉴 바\*\*: 파일, 편집, 테마, 찾기 메뉴가 위치하며, 각 메뉴 클릭 시 하위 메뉴 항목들이 나타납니다.

\* \*\*텍스트 영역\*\*: 스크롤 가능한 주 텍스트 편집 공간입니다.

\* \*\*상태 표시줄\*\*: 하단에 커서의 현재 위치(줄, 열)나 인코딩 같은 간단한 상태 정보를 보여줍니다.

# LLM 과 AI Agent Gen AI와 Vibe Coding

https://gemini.google.com/app/2fd4aa8c43d746b8

Gemini  
2.5 Pro ▾ PRO

GUI 기반 텍스트 편집기를 개발할꺼야. 파이썬과 Tkinter 라이브러리를 이용할꺼야. 메뉴는 파일 메뉴, 편집 메뉴, 테마 선택 메뉴(다크모드 하위 메뉴 포함), 찾기 메뉴로 구성됨. 이 프로그램 개발을 위한 PRD를 UI 스케치를 포함해 md 포맷으로 작성해.

생각하는 과정 표시 ▾

네, 파이썬과 Tkinter를 사용한 GUI 텍스트 편집기 개발을 위한 제품 요구사항 명세서(PRD)를 UI 스케치를 포함하여 작성해 드릴게요.

## Python/Tkinter 기반 GUI 텍스트 편집기 제품 요구사항 명세서 (PRD)

### 1. 개요

본 문서는 파이썬(Python)과 표준 GUI 라이브러리인 티케이인터(Tkinter)를 사용하여 개발될 데스크톱 텍스트 편집기의 요구사항을 정의합니다. 이 애플리케이션은 기본적인 텍스트 편집 기능과 함께 사용자의 편의를

Gemini에게 물어보기

+ 동영상 Deep Research Canvas

Gemini는 인물 등에 관한 정보 제공 시 실수를 할 수 있으니 다시 한번 확인하세요. 개인 정보 보호 및 Gemini

# LLM 과 AI Agent Gen AI와 Vibe Coding

## Gemini CLI

설치: <https://nodejs.org/en/download> 필요. <https://github.com/google-gemini/gemini-cli>  
키 획득: <https://aistudio.google.com/apikey>



# LLM 과 AI Agent Gen AI와 Vibe Coding

## Gemini CLI

명령어: <https://github.com/google-gemini/gemini-cli/blob/main/docs/cli/commands.md>

MCP 확장: <https://github.com/google-gemini/gemini-cli/blob/main/docs/extension.md>

기본 도구: <https://github.com/google-gemini/gemini-cli/blob/main/docs/tools/index.md>

페어 프로그래밍 방법: <https://cloud.google.com/gemini/docs/codeassist/use-agentic-chat-pair-programmer>

The screenshot shows the official Gemini CLI documentation on Google Cloud's website. The main title is "Gemini CLI" with a "Preview" icon. Below the title, there's a "Send feedback" button and a "Was this helpful?" section with upvote and downvote icons. The main content area starts with a note about Pre-GA Offerings Terms and Service Specific Terms. It then describes the Gemini command line interface (CLI) as an open source AI agent that provides access to Gemini directly in your terminal. The text highlights its ability to use a reason and act (ReAct) loop with built-in tools and MCP servers to handle complex tasks like fixing bugs and improving test coverage. It also mentions its versatility as a local utility for various tasks from content generation to problem solving.

gemini --help

-m: gemini model 지정. gemini -m "gemini-1.5-flash"

-p: 단일 프롬프트 실행. gemini -p "three.js" 의 주요 예제 코드를 알려줘 "

-y: 사용자 확인 없이 자동 실행(yolo 모드)

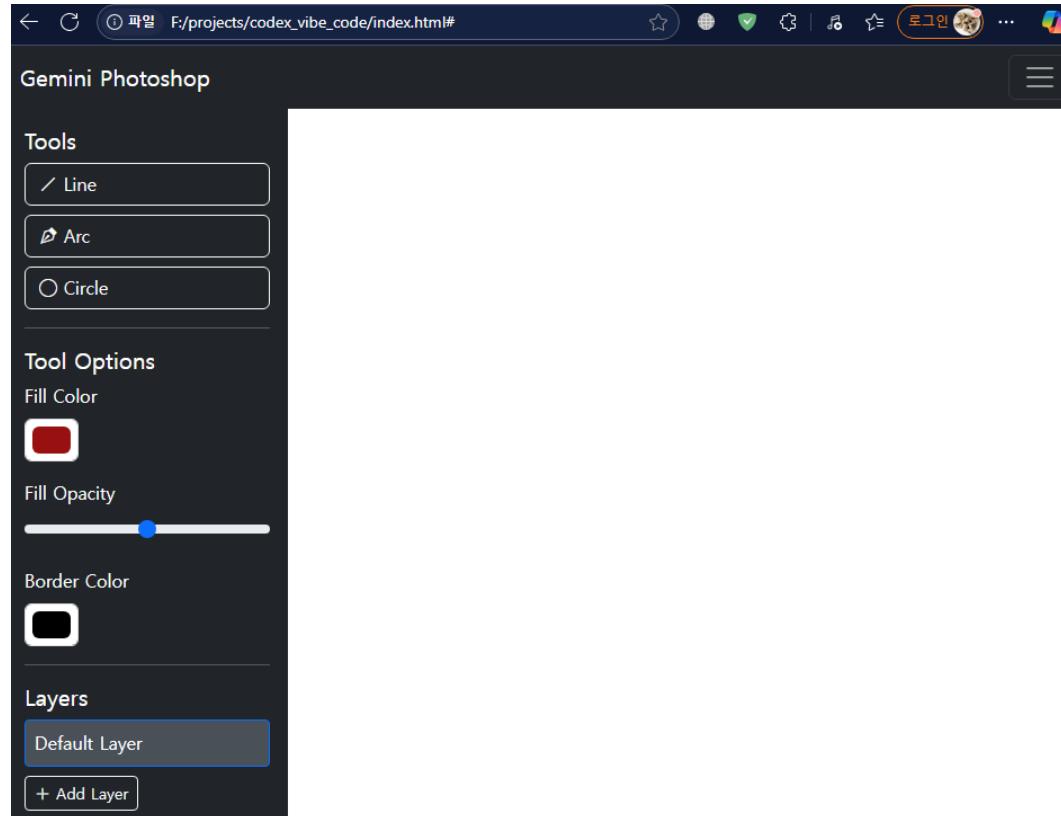
-d: 디버그 옵션. 작업폴더, GEMINI.md 파일 유무, 모델 호출 상세 등 확인 가능

--sandbox: 격리된 환경에서 실행

--show\_memory\_usage: CLI 메모리 사용량 표시

# LLM 과 AI Agent Gen AI와 Vibe Coding

> make photoshop web app using three.js, bootstrap. Menus includes layer, line, arc, circle, fill color with transparent, border color, zoom in/out, pan, download file as JPG



# LLM 과 AI Agent Gen AI와 Vibe Coding

## Codex

OpenAI에서 개발한 바이브 코딩 도구(4월17일 공개). 멀티 AI 에이전트 기능을 지원. 로컬 CLI 모드 지원 단순한 코드 생성에 그치지 않고, 버그 수정, 테스트 실행, 코드 리뷰 제안 등 복잡한 개발 업무를 자동화

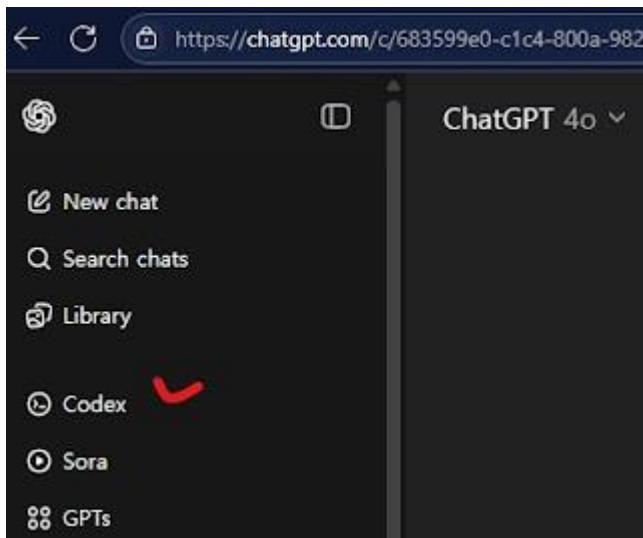
설치: [openai/codex: Lightweight coding agent that runs in your terminal](#)

OpenAI o3 & o4-mini



# LLM 과 AI Agent Gen AI와 Vibe Coding

## Codex



Codex

Environ...

Settings

General

Environments

Search environments

+ Create environment

Name	Repo	Number of tasks
mac999/AI_agent_simple_function_call	mac999/AI_agent_simple_function_call	0

Daddy Makers:  
OpenAI 바이브 코딩  
지원 멀티 에이전트  
Codex 도구 사용법

What are we coding?

Describe a task

mac999/AI\_agent\_simpl... main

mac999/AI\_agent\_simple\_function\_call

Manage environments

57 min ago · mac999/AI\_agent\_simple\_function\_call

Create AI agent with ReAct and OpenAI API

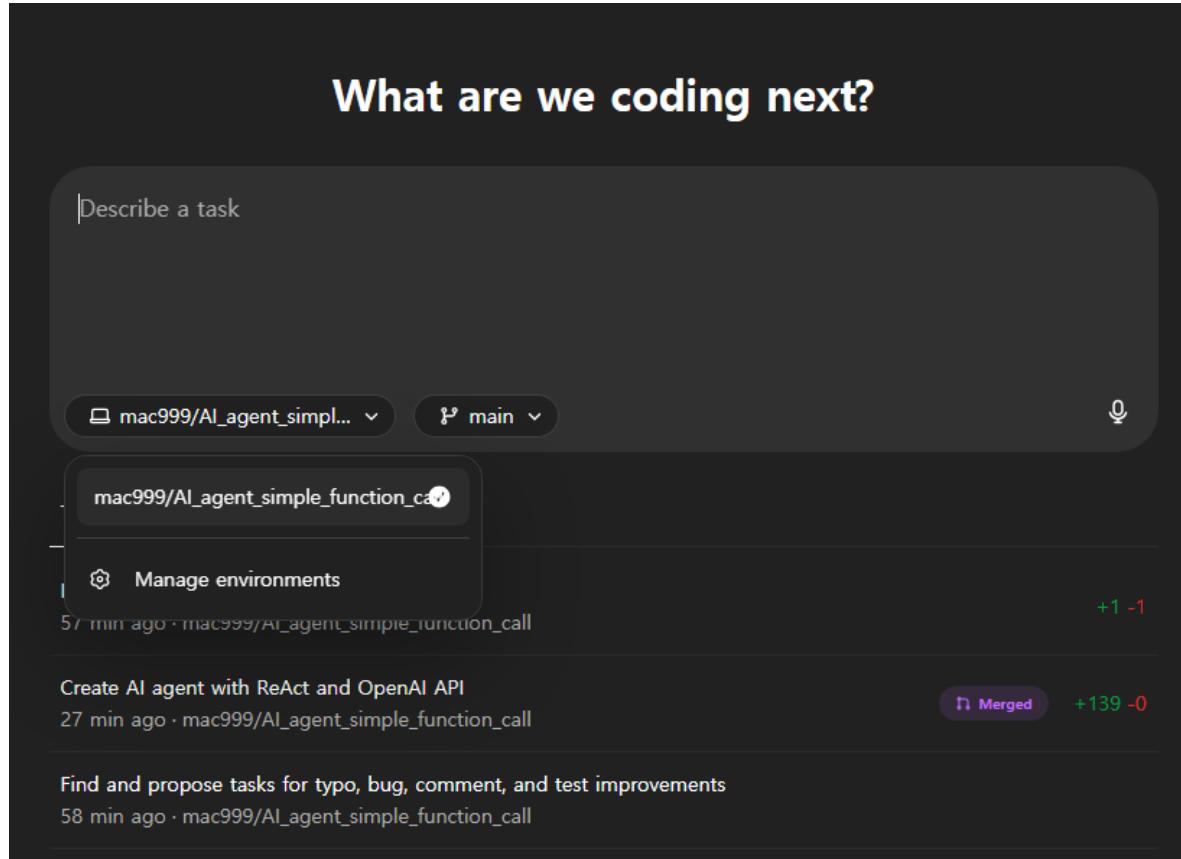
27 min ago · mac999/AI\_agent\_simple\_function\_call

Find and propose tasks for typo, bug, comment, and test improvement

58 min ago · mac999/AI\_agent\_simple\_function\_call

# LLM 과 AI Agent Gen AI와 Vibe Coding

## Codex



make AI agent simple source code using python to implement ReAct, with openai api. it supports CLI for singeline query mode, chatbot mode.

Worked for 1m 34s >

#### Notes

- Added `react_agent.py` containing a minimal ReAct-style loop that uses OpenAI function calling for searching and calculations.
- Updated `README.md` with instructions on how to run the new CLI tool.

#### Summary

- Documented the usage of the new ReAct CLI in the README, including single-query and chatbot modes ⓘ
- Implemented `react_agent.py` to provide search and calculator tools and handle interactive or one-shot prompts ⓘ

#### Testing

- ✖ `pytest -q` (no tests were found) ↗

#### Files (2)

README.md +19 -0

react\_agent.py New



Request changes or ask a question

Diff Logs

react\_agent.py New

```
54 + }
55 +
56 +
57 + def react_loop(messages):
58 +     """Run the ReAct reasoning loop until a final answer is produced."""
59 +     while True:
60 +         response = openai.ChatCompletion.create(
61 +             model="gpt-4o",
62 +             messages=messages,
63 +             tools=tools,
64 +             tool_choice="auto",
65 +         )
66 +         msg = response["choices"][0]["message"]
67 +         if msg.get("tool_calls"):
68 +             messages.append(msg)
69 +             for call in msg["tool_calls"]:
70 +                 name = call["function"]["name"]
71 +                 args = json.loads(call["function"]["arguments"])
72 +                 result = FUNCTION_MAP[name](**args)
73 +                 messages.append({
74 +                     "role": "tool",
75 +                     "tool_call_id": call["id"],
76 +                     "content": result,
77 +                 })
78 +             continue
79 +         messages.append(msg)
80 +         return msg["content"]
81 +
82 +
83 + def single_query(prompt: str) -> str:
84 +     messages = [
85 +         {"role": "system", "content": "You are a helpful assistant using ReAct."},
86 +         {"role": "user", "content": prompt},
87 +     ]
88 +     return react_loop(messages)
89 +
90 +
91 + def chat_mode():
92 +     messages = [{"role": "system", "content": "You are a helpful assistant using ReAct."}]
93 +     print("Starting chat. Type 'exit' to quit.")
94 +     while True:
95 +         user = input("> ")
96 +         if user.lower() in {"exit", "quit"}:
```

# LLM 과 AI Agent Gen AI와 Vibe Coding

## Codex

The screenshot shows a code editor interface with the following details:

- File Explorer:** On the left, it shows a project structure under "OPEN EDITORS". The file "react\_agent.py" is currently selected.
- Code Editor:** The main area displays Python code for an AI agent. The code includes functions for handling user input, managing messages, and running a main loop.
- Bottom Status Bar:** Shows the author's name (Tae woong), the file name (main), line count (Ln 107), column count (Col 31), and encoding (UTF-8).

```
def single_query(prompt: str) -> str:
    messages = [
        {"role": "system", "content": "You are a helpful assistant using ReAct."},
        {"role": "user", "content": prompt},
    ]
    return react_loop(messages)

def chat_mode():
    messages = [{"role": "system", "content": "You are a helpful assistant using ReAct."}]
    print("Starting chat. Type 'exit' to quit.")
    while True:
        user = input("> ")
        if user.lower() in {"exit", "quit"}:
            break
        messages.append({"role": "user", "content": user})
        answer = react_loop(messages)
        print(answer)

def main():
    parser = argparse.ArgumentParser(description="Simple ReAct agent")
    parser.add_argument("prompt", nargs="?", help="Single prompt to run")
    parser.add_argument("-i", "--interactive", action="store_true", help="Chatbot mode")
    args = parser.parse_args()

    openai.api_key = os.environ.get("OPENAI_API_KEY")

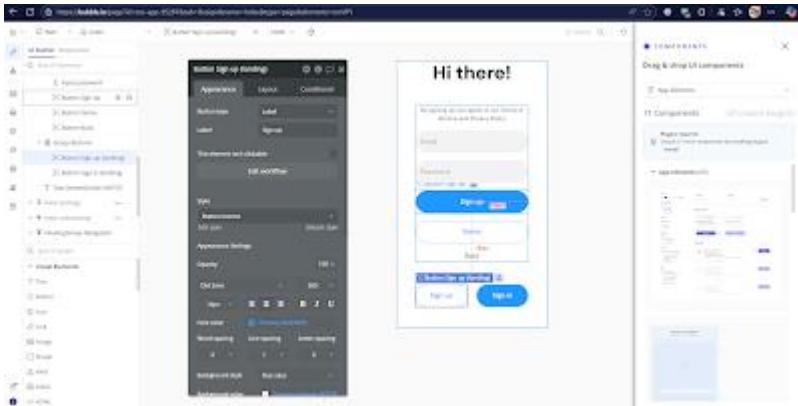
    if args.interactive:
        chat_mode()
    elif args.prompt:
        print(single_query(args.prompt))
    else:
        parser.print_help()

if __name__ == "__main__":
    main()
```

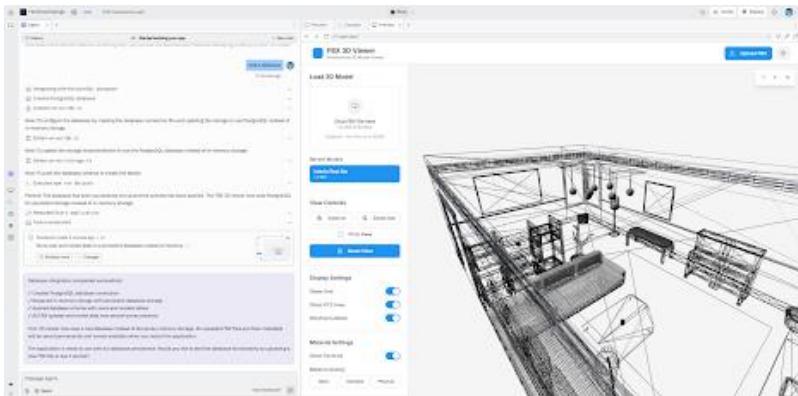
# LLM 과 AI Agent Gen AI와 Vibe Coding

## 노코드(No-code)

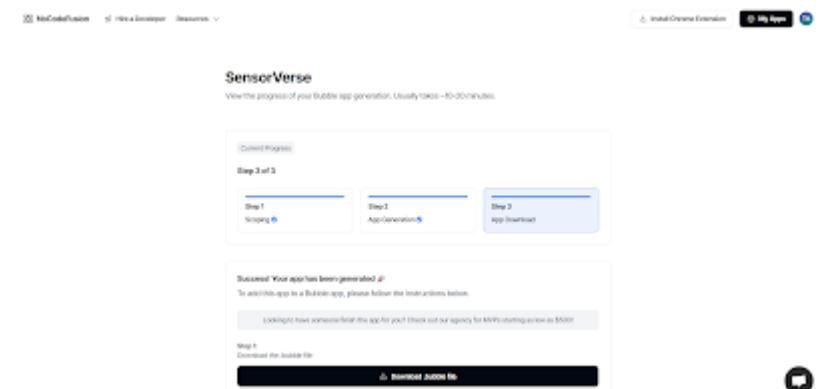
Bubble.io: 거의 모든 종류의 웹 앱을 만들 수 있는 높은 유연성과 기능을 제공



Replit: 코드를 작성, 실행, 공유하는 과정을 웹에서 편리하게 수행 가능한 IDE



Nocodefusion: 생성형 AI 기술과 노코드 플랫폼을 결합하여 AI 기반 앱 신속 개발에 중점을 둔 도구



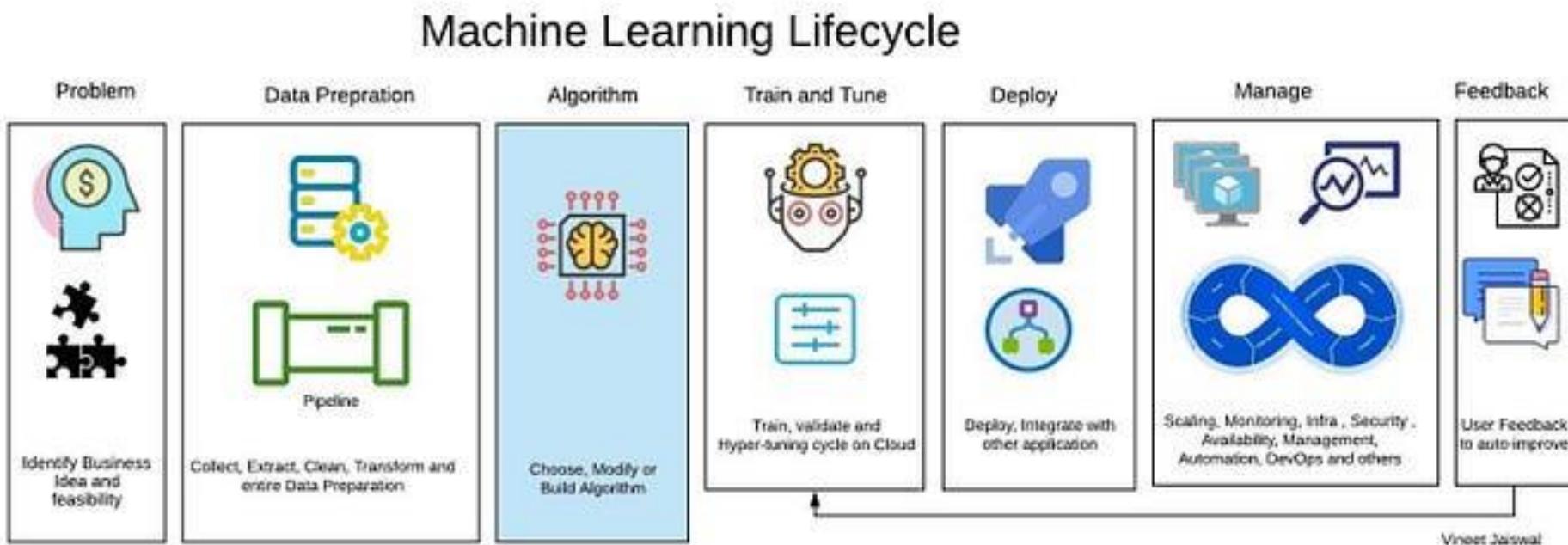
Prompt: create FBX 3D viewer web app using django, bootstrap, threejs. the app includes view menu such as zoom in/out, extent, rotate. the app's canvas renders grid, xyz mark defaultly

# AI 모델의 서비스화 개요

모델 학습 vs 서빙  
API 기반 AI 서비스 아키텍처 소개  
ONNX, FastAPI, Flask 등 경량화 및 배포 전략 개요

# AI 모델의 서비스화

## 모델 학습 vs 서빙

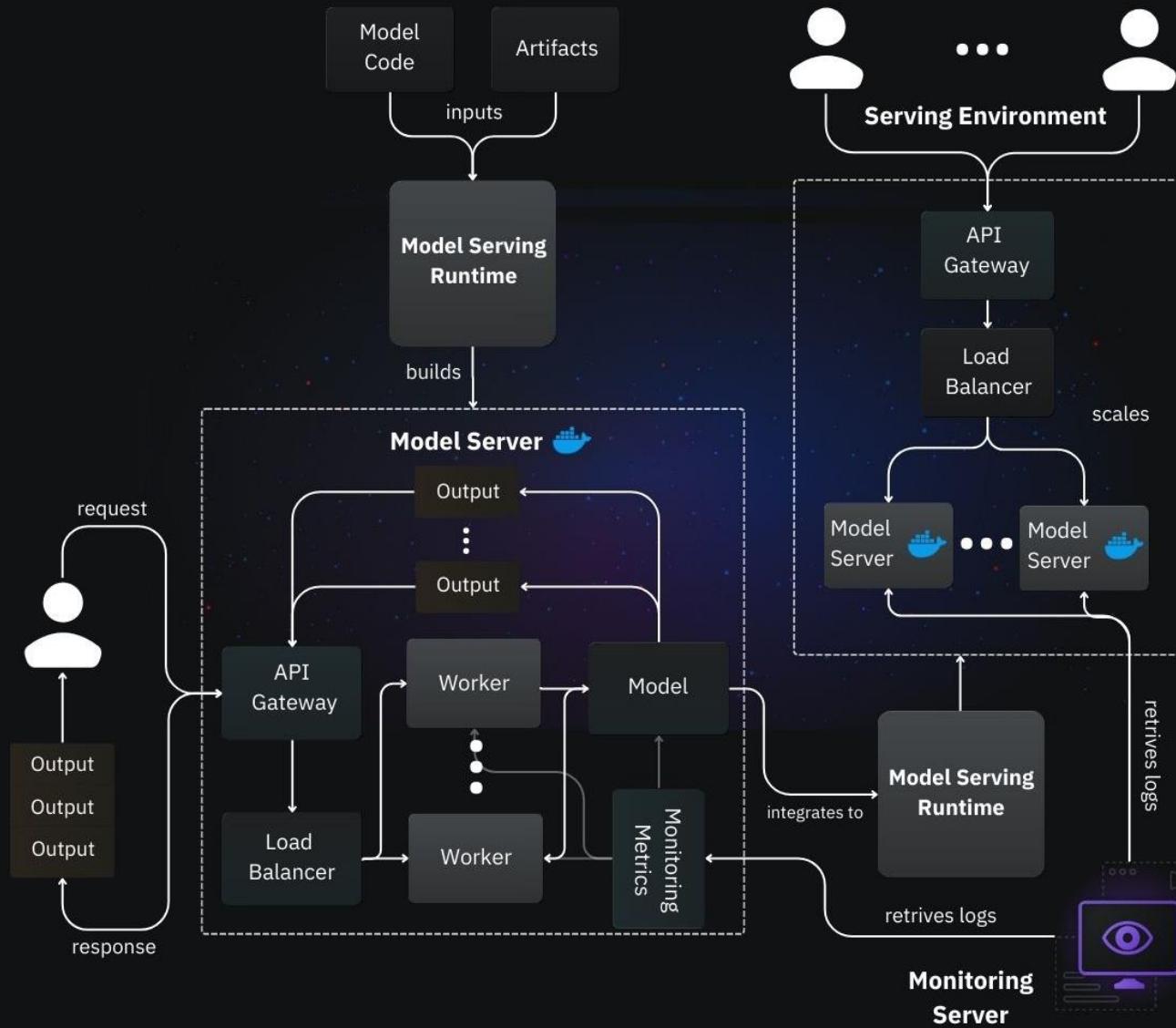


[Introduction & Comparison of MLOps platforms: AWS Sagemaker, Azure Machine Learning, GCP Vertex AI | by Vineet Jaiswal | Medium](#)

# AI 모델의 서비스화

모델 학습 vs 서빙

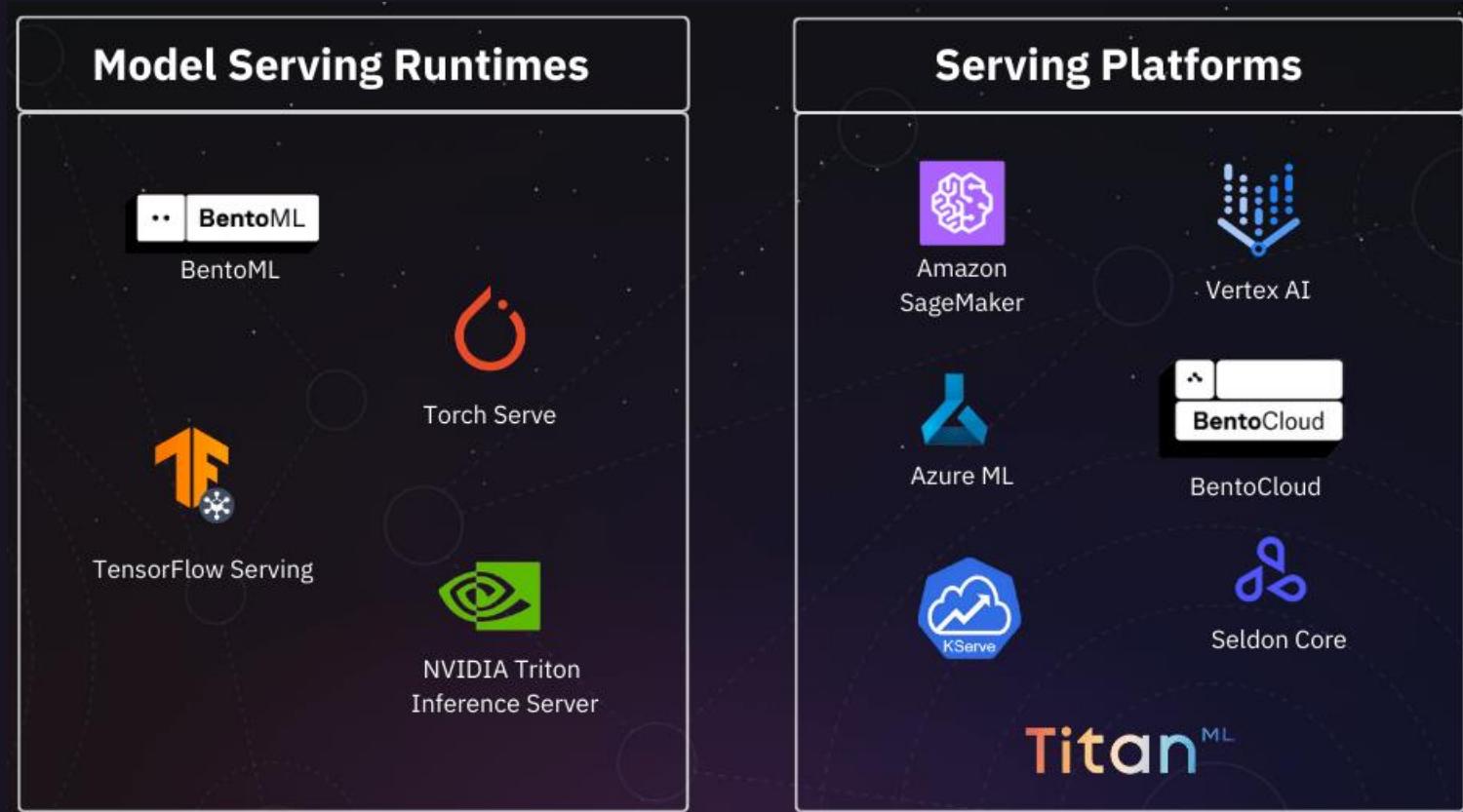
## Model Serving Runtime & Serving Platform



Best Tools  
For ML  
Model  
Serving

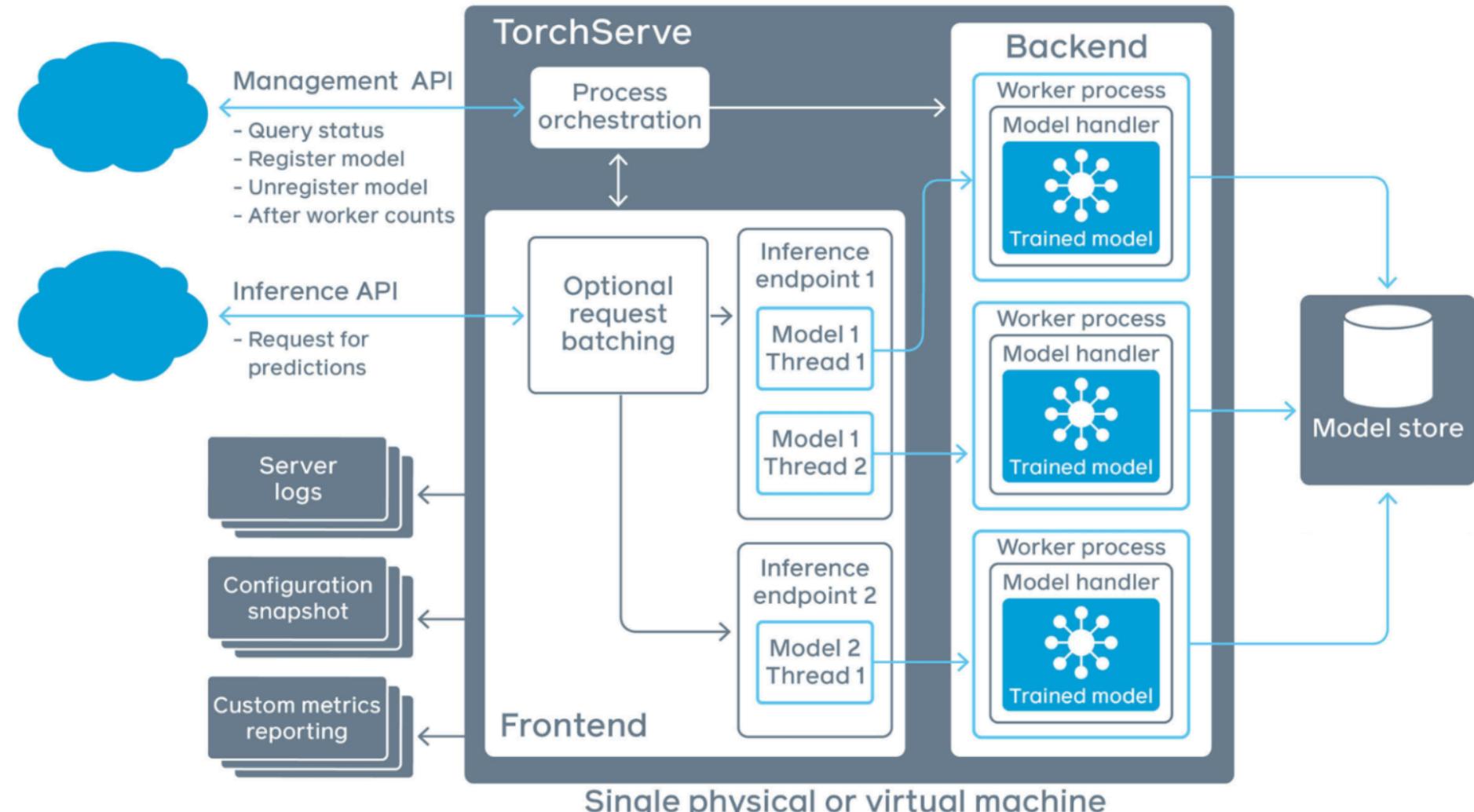
# AI 모델의 서비스화

모델 학습 vs 서빙



# AI 모델의 서비스화

모델 학습 vs 서빙



[TorchServe — PyTorch/Serve master documentation](#)

Best Tools  
For ML  
Model  
Serving

# AI 모델의 서비스화 모델 학습 vs 서빙

## Takeoff Playground

Select a Model Group: primary

What are the main ingredients in a cake?

The main ingredients in a cake are flour, sugar, eggs, baking powder, butter, and milk. These ingredients form the base of most cakes, with variations and additional ingredients depending on the specific type of cake being made.

Send 🚀

### Generation Config

System Prompt

Temperature: 1

Sampling top p: 1

Sampling top k: 10

Repetition penalty: 1

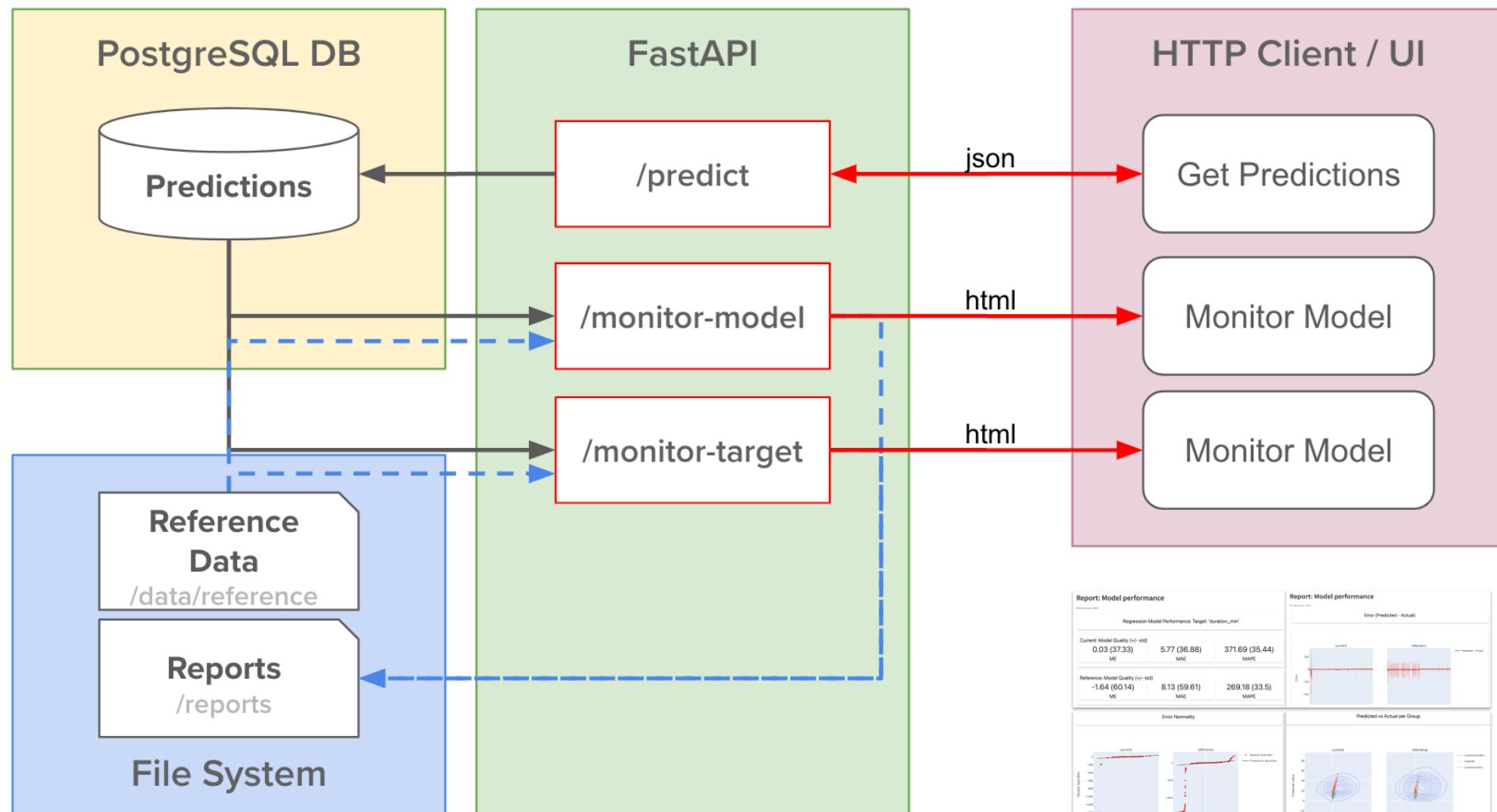
Max generation length: 300

No repeat ngram size: 0

Regex String: Regex string

Json Schema: Json Schema

# AI 모델의 서비스화 API 기반 AI 서비스 아키텍처 소개

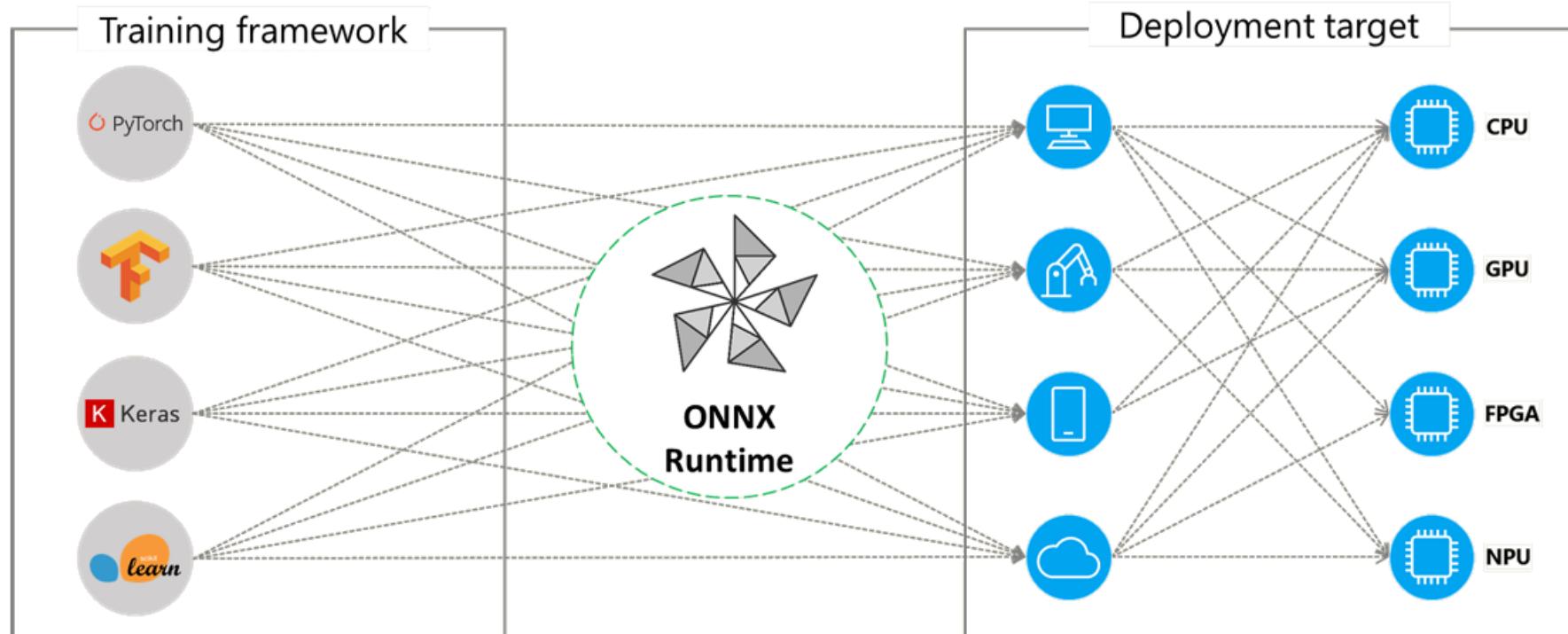


# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

## ONNX(Open Neural Network Exchange)

ONNX는 표준화된 딥 러닝 모델 저장 및 관리 형식. 표준화된 딥러닝 그래프 수행 계산, 입출력 및 텐서 형식 제공. 딥러닝 프레임워크, 다양한 언어 간 모델 입출력 호환성 제공. GPU 뿐 아니라 다양한 TPU 호환성 제공



# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

## Create

### Frameworks



### Services



Converters

Native support

Native support

## ONNX Model

## Deploy

### Azure

- Azure Machine Learning services
- Ubuntu VM
- Windows Server 2019 VM

### Windows Devices

Native support

Converters

### Other Devices (iOS, etc)

pip install onnx onnxruntime-gpu

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

```
import os, torch, onnxruntime, numpy as np

# Defining PyTorch model
class MyModel(torch.nn.Module):
    def __init__(self):
        super(MyModel, self).__init__()
        self.fc1 = torch.nn.Linear(10, 512)
        self.fc2 = torch.nn.Linear(512, 1024)
        self.fc3 = torch.nn.Linear(1024, 512)
        self.fc4 = torch.nn.Linear(512, 256)
        self.fc5 = torch.nn.Linear(256, 10)

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = torch.relu(self.fc2(x))
        x = torch.relu(self.fc3(x))
        x = torch.relu(self.fc4(x))
        x = self.fc5(x)
        return x

# Creating an instance
model = MyModel()

# Training data
X_train = torch.randn(1000, 10)
y_train = torch.randn(1000, 10)

# Training loop
optimizer = torch.optim.Adam(model.parameters(),
lr=0.01)
criterion = torch.nn.MSELoss()
```

```
for epoch in range(200):
    optimizer.zero_grad()
    outputs = model(X_train)
    loss = criterion(outputs, y_train)
    loss.backward()
    optimizer.step()

    if epoch % 20 == 0:
        print(f'Epoch {epoch}, Loss: {loss.item():.4f}')

# Defining input example
example_input = torch.randn(1, 10)

# Exporting to ONNX format
module_path =
os.path.dirname(os.path.abspath(__file__))
model_path = os.path.join(module_path, "model.onnx")
torch.onnx.export(model, example_input, model_path)

# Using ONNX model
session = onnxruntime.InferenceSession(model_path)
input_name = session.get_inputs()[0].name
onnx_output = session.run(None, {input_name:
example_input.numpy()})
```

```
Epoch 120, Loss: 0.9953
Epoch 140, Loss: 0.9953
-0.14417903 -0.27374357 0.04801298 -0.05633638]]
ONNX Output: [[-0.02958158 -0.07041439 0.16391063 -
-0.14417902 -0.27374354 0.04801297 -0.05633637]]
```

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

```
# Quantizing the ONNX model
from onnxruntime.quantization import quantize_dynamic, QuantType

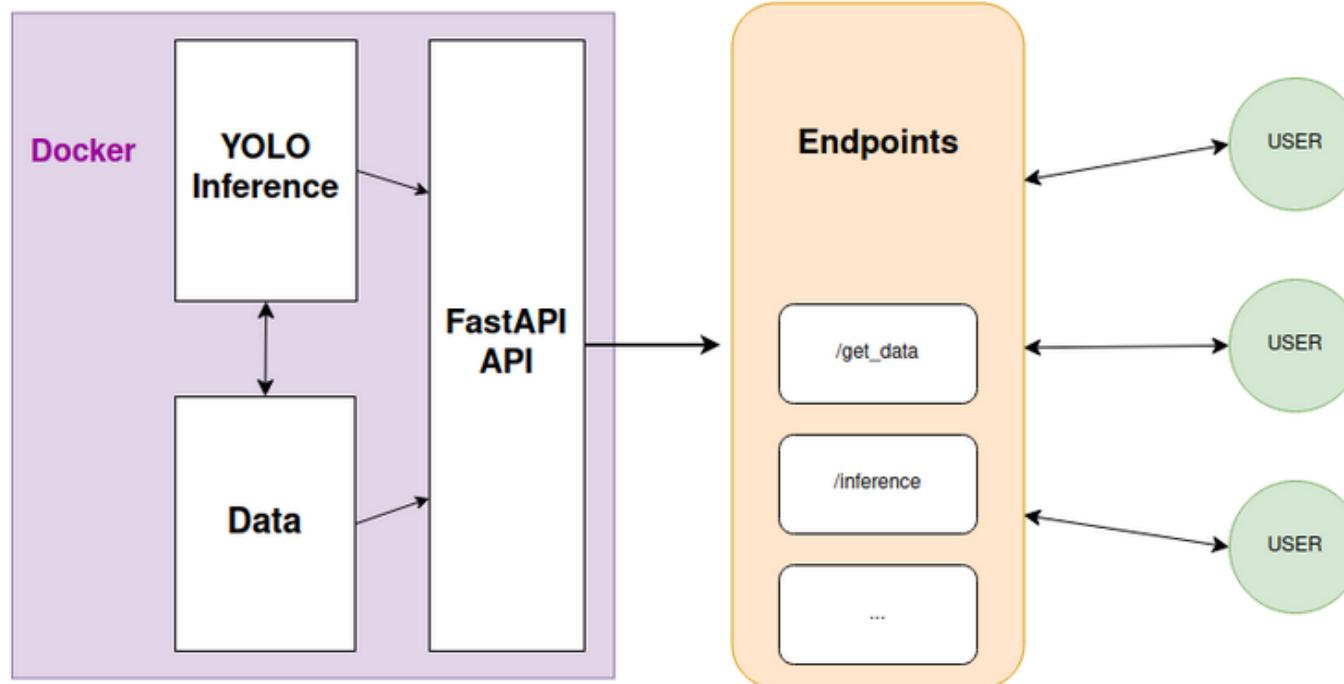
quantized_model_path = os.path.join(module_path, "model_quantized.onnx")
quantize_dynamic(model_path, quantized_model_path, weight_type=QuantType.QUInt8)

# Load the quantized model
quantized_session = onnxruntime.InferenceSession(quantized_model_path)
quantized_output = quantized_session.run(None, {input_name: example_input.numpy()})
print("Quantized ONNX Output:", quantized_output[0])

# Compare file sizes
original_size = os.path.getsize(model_path)
quantized_size = os.path.getsize(quantized_model_path)
print(f"Original model size: {original_size / 1024:.2f} KB")
print(f"Quantized model size: {quantized_size / 1024:.2f} KB")
print(f"Size reduction: {(1 - quantized_size/original_size)*100:.1f}%")
```

```
Quantized ONNX Output: [[ 0.7099464
    -0.5664326 -0.25450945 -1.1977834
Original model size: 4648.18 KB
Quantized model size: 1174.10 KB
Size reduction: 74.7%
```

## FastAPI server with YOLO



[Daddy Makers: FastAPI, Uvicorn, Websocket 기반 Open API 서버 개발](#)

[YOLOv8-docker/README.md at main · JavierMtz5/YOLOv8-docker](#)

[YOLO inference with Docker via API | Towards Data Science](#)

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

```
import os, json
from ultralytics import YOLO
from fastapi import FastAPI, UploadFile, File, Query
from typing import List, Dict, Any, Union
from PIL import Image
from io import BytesIO

app = FastAPI()

model_path = os.path.join(os.path.dirname(__file__), "yolov8m.pt")
model = YOLO(model_path)

def inference_on_path(imgs_path: str) -> List[Dict[str, Any]]:
    results: ultralytics.engine.results.Results = model(source=imgs_path, show=False, conf=0.45)
    results_data = []
    for result in results:
        result_metadata = {
            'shape': result.orig_shape,
            'path': result.path,
            'detections': json.loads(result.tojson())
        }
        results_data.append(result_metadata)

    return results_data

def inference_on_img(img: Image) -> List[Dict[str, Any]]:
    results: ultralytics.engine.results.Results = model(source=img, show=False, conf=0.45)
    result_data = json.loads(results[0].tojson())

    return result_data
```

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

```
import os, json
from ultralytics import YOLO
from fastapi import FastAPI, UploadFile, File, Query
from typing import List, Dict, Any, Union
from PIL import Image
from io import BytesIO

app = FastAPI()

...
@app.post('/detect_img')
async def detect_img(
    img: UploadFile = File(...),
) -> Dict[str, Union[int, Dict[str, Any]]]:
    if not img:
        return {'status_code': 400, 'data': {'message': 'No upload file sent'}}

    img_content = await img.read()
    image_stream = BytesIO(img_content)
    image = Image.open(image_stream)

    inference_results_data = inference_on_img(img=image)
    return {'status_code': 200,
            'data': {
                'image_name': img.filename,
                'image_size': img.size,
                'inference_results': inference_results_data
            }
    }
```

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

## FastAPI server with YOLO

FastAPI 0.1.0 OAS 3.1

/openapi.json

### default

GET	/ Home	▼
GET	/detect Detect	▼
POST	/detect_img Detect Img	▼
GET	/images Get Available Images	▼

Schemas

Body\_detect\_img\_detect\_img\_post > Expand

HTTPValidationError > Expand all object

ValidationError > Expand all object

GET /detect Detect

Runs YOLO inference for all the images available in the path received as query parameter

Parameters

Try it out

Name	Description
path <small>* required string (query)</small>	Path to the images directory

path

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

## FastAPI server with YOLO

```
echo 3. Path-based Image Analysis (Current Folder)
curl -X GET "%API_BASE%/detect?path=cat.jpg" -H "Content-Type: application/json"

echo.
if exist "cat.jpg" (
    echo 4. Image Upload Test (cat.jpg)
    curl -X POST "%API_BASE%/detect_img" -F "img=@cat.jpg"
```

```
F:\projects\AI_foundation_tutorial\8_serving\fastapi_model_serving>test_api.bat
Object Detection Server API Test Start (curl)

1. Health Check
{"status_code":200,"data":{"api_health_status":"OK"}}

2. List Images in Current Folder
{"status_code":200,"data":{"available_images":["cat.jpg"],"size":1}}


3. Path-based Image Analysis (Current Folder)
{"status_code":200,"data":{"inference_results":[{"shape":[1536,2048],"path":"F:\\projects\\AI_foundation_tutorial\\8_serving\\fastapi_model_serving\\cat.jpg","detections":[{"name":"cat","class":15,"confidence":0.96055,"box":{"x1":3.84907,"y1":348.77158,"x2":1634.66895,"y2":1534.54602}]}]}}

4. Image Upload Test (cat.jpg)
{"status_code":200,"data":{"image_name":"cat.jpg","image_size":322451,"inference_results":[{"name":"cat","class":15,"confidence":0.96055,"box":{"x1":3.84907,"y1":348.77158,"x2":1634.66895,"y2":1534.54602}]}}

Test Completed!

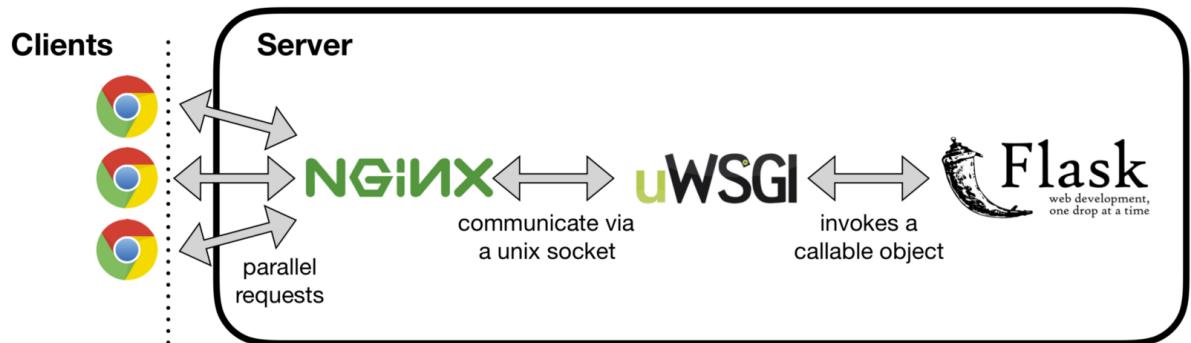
F:\projects\AI_foundation_tutorial\8_serving\fastapi_model_serving>
```

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

Flask

pip install Flask



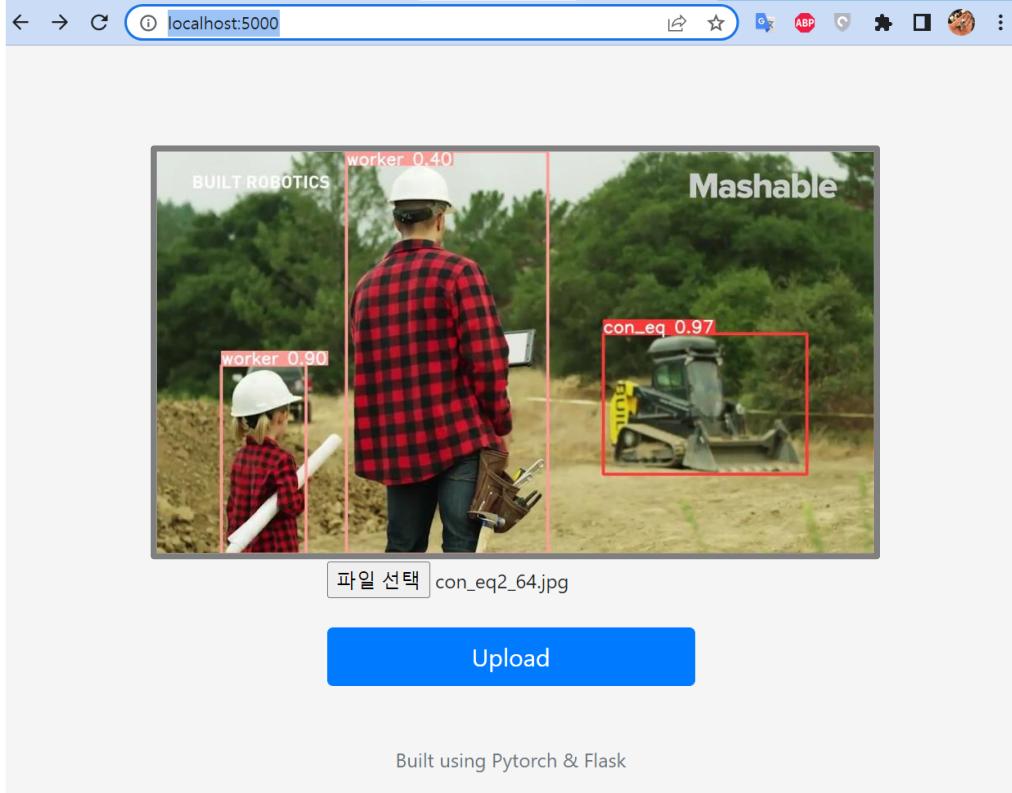
```
main.py  ×

 1  # main.py
 2
 3  from flask import Flask
 4
 5  app = Flask(__name__)
 6
 7  @app.route("/")
 8  def index():
 9      return "Congratulations, it's a web app!"
10
11 if __name__ == "__main__":
12     app.run(host="127.0.0.1", port=8080, debug=True)
13
14
```

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

## Flask



projects > notebook > 6\_yolo\_practice > yolov5\_flask

이름	수정한 날
.git	2022-10-
.github	2022-10-
__pycache__	2022-10-
docs	
models	
static	
templates	
tests	
utils	
con_eq2_69.jpg	
README.md	
a_file.png	
yolov5_con.pt	
yolov5s.pt	
export.py	
get-pip.py	
hubconf.py	
restapi.py	
webapp.py	
requirements.txt	
openapi_test.bat	2022-10-
run.bat	2022-10-
dataset_construct_worker.yaml	2022-09-
.gitignore	2022-09-
Dockerfile	2022-09-

A screenshot of a web browser showing a large number of excavators in a parking lot, each detected by a red bounding box. The URL is `localhost:5000/static/predict.png`.

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

## Flask

Curl utility installation <https://curl.se/download.html>

python restapi.py

curl -X POST -F image=@con\_eq2\_69.jpg "http://127.0.0.1:5000/v1/object\_detection"

```
(venv_tf2) C:\#project\st\yolov5_flask>python restapi.py
YOLOv5 1.3-2-gac18396 Python-3.9.12 torch-1.12.1+cu113 CUDA:0 (NVIDIA GeForce RTX 3080 Laptop GPU, 8192MiB)
Fusing layers...
Model summary: 213 layers, 7015519 parameters, 0 gradients, 15.8 GFLOPs
Adding AutoShape...
* Serving Flask app 'restapi'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.30.1.88:5000
Press Ctrl+C to quit
127.0.0.1 - - [16/Oct/2022 08:02:03] "GET / HTTP/1.1" 404 -parameter Tuning
127.0.0.1 - - [16/Oct/2022 08:02:05] "GET / HTTP/1.1" 404 -
127. 명령 프롬프트
127. 2022-10-16 오전 08:07          81 openapi.bat
127. 2022-10-09 오후 08:10          18 run.bat
127.          2개 파일           99 바이트
127.          0개 디렉터리   179,548,721,152 바이트 남음
127.          0개 디렉터리   163,751,663,616 바이트 남음
127. C:\#projects\#yolov5_flask>openapi.bat
C:\#projects\#yolov5_flask>curl -X POST -F image=@con_eq2_69.jpg "http://127.0.0.1:5000/v1/object_detection"
[{"xmin":24.6365852356,"ymin":561.6345214844,"xmax":129.339263916,"ymax":720.0,"confidence":0.9224505424,"class":1,"name":"worker"}, {"xmin":526.0735473633,"ymin":33.5504608154,"xmax":732.2012329102,"ymax":541.9304199219,"confidence":0.9142798185,"class":0,"name":"con_eq"}, {"xmin":1021.1021728516,"ymin":352.8305053711,"xmax":1228.3970947266,"ymax":514.4751586914,"confidence":0.910230875,"class":0,"name":"con_eq"}, {"xmin":24.6365852356,"ymin":561.6345214844,"xmax":129.339263916,"ymax":720.0,"confidence":0.9224505424,"class":1,"name":"worker"}, {"xmin":526.0735473633,"ymin":33.5504608154,"xmax":732.2012329102,"ymax":541.9304199219,"confidence":0.9142798185,"class":0,"name":"con_eq"}, {"xmin":1021.1021728516,"ymin":352.8305053711,"xmax":1228.3970947266,"ymax":514.4751586914,"confidence":0.910230875,"class":0,"name":"con_eq"}, {"xmin":256.1762390137,"ymin":964.9847412109,"xmax":316.9440612793,"confidence":0.8944447041,"class":0,"name":"con_eq"}, {"xmin":850.4646606445,"ymin":253.5209350586,"xmax":895.5513305664,"ymax":320.7725830078,"confidence":0.8598570824,"class":0,"name":"con_eq"}, {"xmin":983.1149291992,"ymin":266.1564941406,"xmax":1023.3728637695,"ymax":318.2346801758,"confidence":0.8123363853,"class":0,"name":"con_eq"}, {"xmin":691.6927490234,"ymin":109.5151672363,"xmax":793.2215576172,"ymax":394.5242614746,"confidence":0.7660834789,"class":0,"name":"con_eq"}, {"xmin":978.7466430664,"ymin":314.1014709473,"xmax":1081.8812255859,"ymax":403.5487365723,"confidence":0.7509086728,"class":0,"name":"con_eq"}, {"xmin":643.0079345703,"ymin":246.8774414062,"xmax":674.1931152344,"ymax":303.2550048828,"confidence":0.6875187159,"class":0,"name":"con_eq"}, {"xmin":964.1707763672,"ymin":258.235209350586,"xmax":993.1149291992,"ymax":319.0886535645,"confidence":0.604162395,"class":0,"name":"con_eq"}, {"xmin":755.126159668,"ymin":161.3591308594,"xmax":803.8353881836,"ymax":321.1946411133,"confidence":0.5691390634,"class":0,"name":"con_eq"}, {"xmin":1163.9505615234,"ymin":320.2859191895,"xmax":1243.7830810547,"ymax":410.7705993652,"confidence":0.428168267,"class":0,"name":"con_eq"}, {"xmin":867.5435791016,"ymin":246.4231872559,"xmax":906.6706542969,"ymax":315.4702453613,"confidence":0.428168267,"class":0,"name":"con_eq"}, {"xmin":967.5684204102,"ymin":314.8999938965,"xmax":1001.7764282227,"ymax":360.8254699707,"confidence":0.3965543807,"class":0,"name":"con_eq"}, {"xmin":1033.970703125,"ymin":237.0289916992,"xmax":1096.1882324219,"ymax":337.1311645508,"confidence":0.3563090563,"class":0,"name":"con_eq"}, {"xmin":1242.9893798828,"ymin":337.2352294922,"xmax":1280.0,"ymax":457.3475952148,"confidence":0.3033725619,"class":0,"name":"con_eq"}]
```

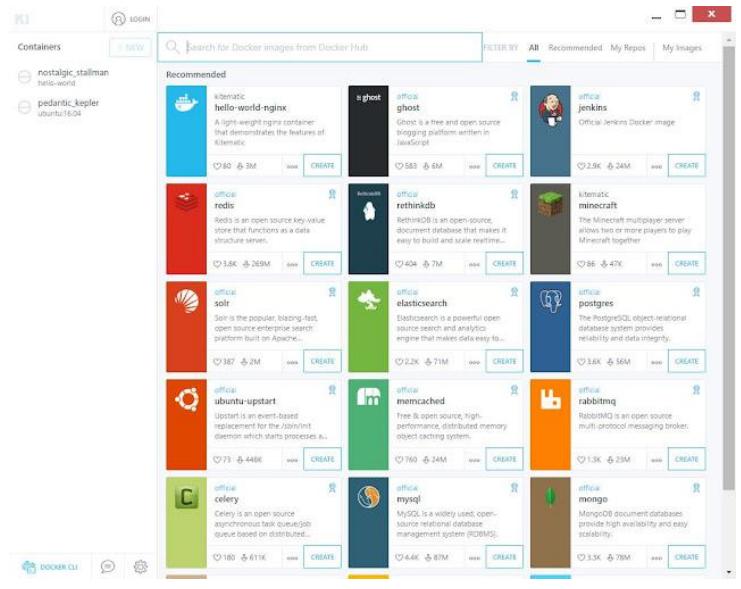
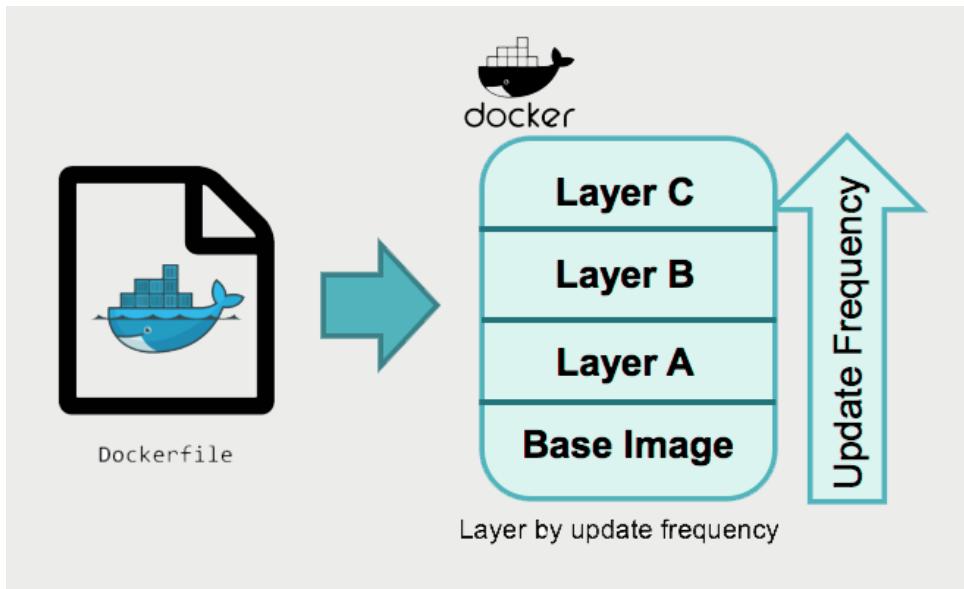
# AI 모델의 서비스화

## ONNX, FastAPI, Flask 등 경량화 및 배포 전략

### Flask

도커는 컨테이너 기반 오픈소스 가상화 플랫폼. 실행 중인 컨테이너에 접속해 명령을 실행하고, apt-get 으로 패키지를 설치할 수 있음. 여러 프로세스를 백그라운드로 실행

도커는 2013년 3월 Pycon 컨퍼런스(산타클라라)에서 dotCloud 창업자인 Solomon Hykes가 리눅스 컨테이너의 미래라는 세션을 발표하면서 처음 알려짐. 이후 도커는 급성장해 2015년 4월 1,100억원 투자유치를 받았으며, 마이크로소프트가 2016년 6월 4 billion dollar에 도커를 인수하려 함



# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

```
# Use Python 3.11 slim image as base
FROM python:3.11-slim

# Set working directory and environment variables
WORKDIR /app
ENV PYTHONUNBUFFERED=1
ENV PYTHONDONTWRITEBYTECODE=1

# Install minimal system dependencies for OpenCV and YOLO
RUN apt-get update && \
    apt-get install -y --no-install-recommends \
    curl \
    libglib2.0-0 \
    ...
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY . .

# Create a non-root user
RUN useradd --create-home --shell /bin/bash app \
    && chown -R app:app /app
USER app

# Expose port 8000
EXPOSE 8000

# Command to run the application
CMD ["fastapi", "run", "server.py", "--host", "0.0.0.0", "--port", "8000"]
```



fastapi[standard]>=0.104.0  
uvicorn[standard]>=0.24.0  
ultralytics>=8.0.0  
opencv-python>=4.8.0  
Pillow>=10.0.0  
python-multipart>=0.0.6

# AI 모델의 서비스화

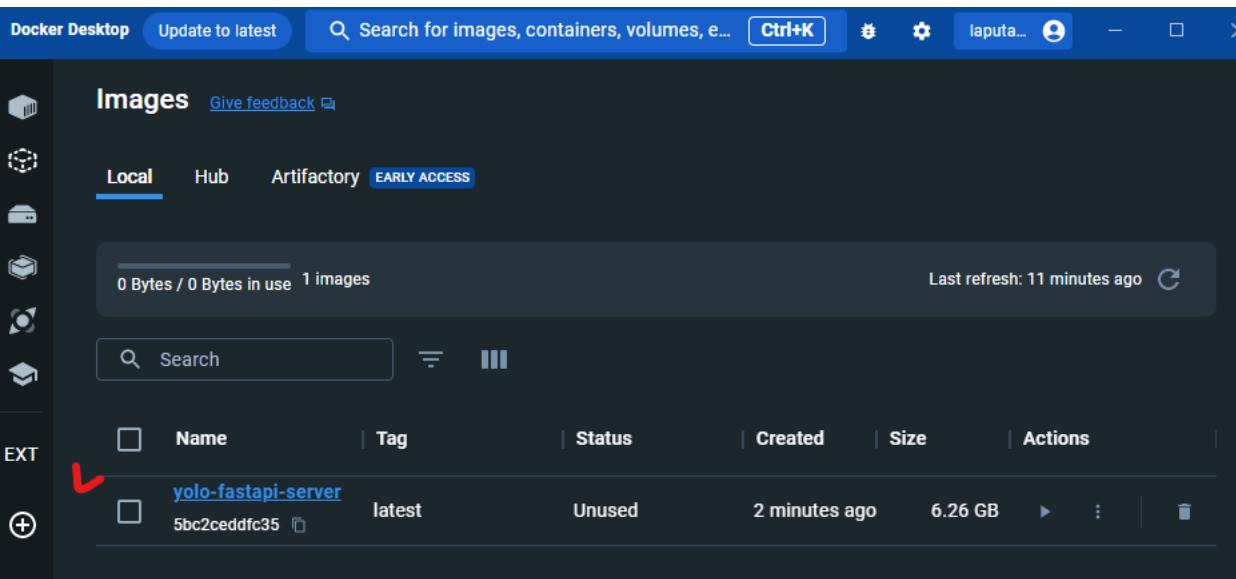
ONNX, FastAPI, Flask 등 경량화 및 배포 전략

```
echo Building Docker image...
docker build -t yolo-fastapi-server .
```

```
[+] Building 61.6s (9/12)
=> [internal] load build context
=> => transferring context: 1.29kB
=> CACHED [2/7] WORKDIR /app
=> [3/7] RUN apt-get update && apt-get install -y --no-install-recommends
=> [4/7] COPY requirements.txt .
=> [5/7] RUN pip install --no-cache-dir -r requirements.txt
=> => # 0:00
=> => # Downloading torch-2.7.1-cp311-cp311-manylinux_2_28_1-x86_64.whl
... 82%
```

```
docker:default
0.0s
0.0s
0.0s
4.9s
0.0s
```

Container started successfully!  
API is available at: <http://localhost:8000>  
API Documentation: <http://localhost:8000/docs>



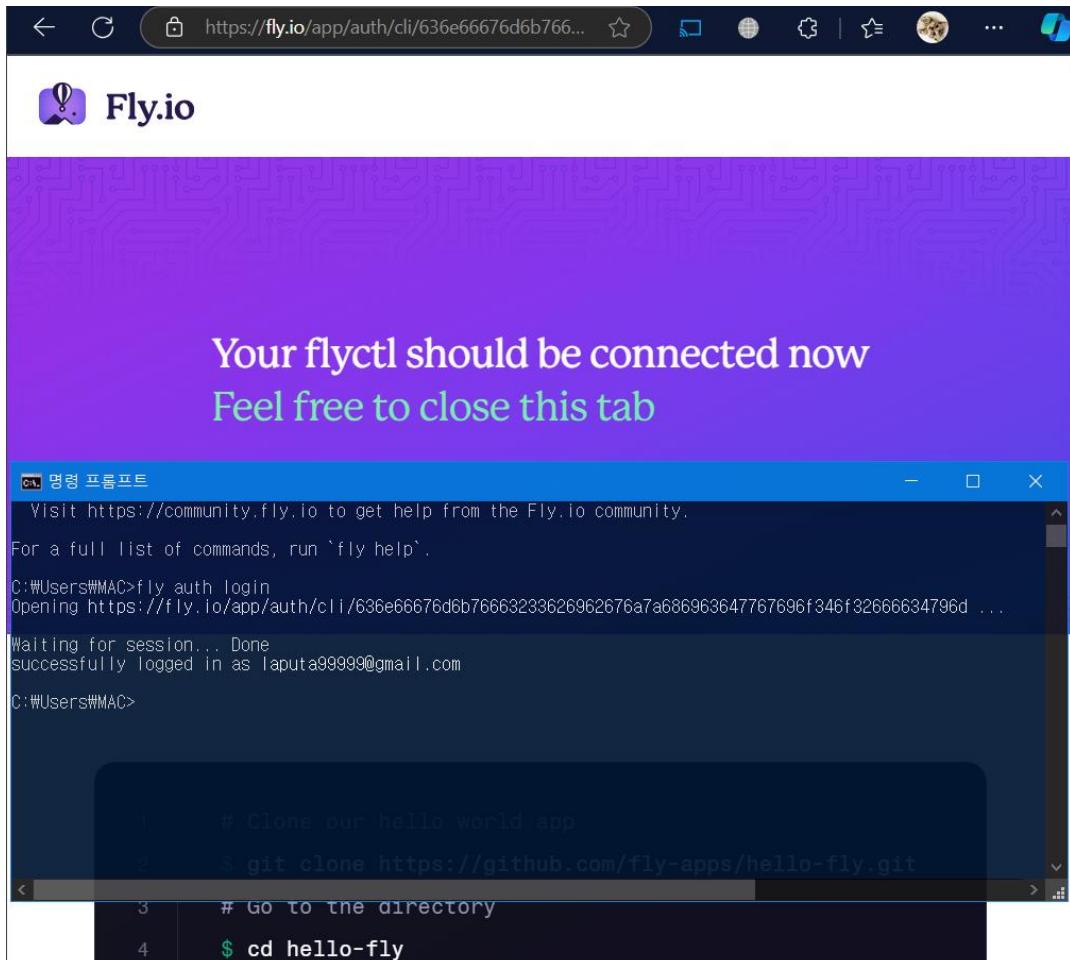
Serve your first model with Scikit-Learn +  
Flask + Docker | by Carl W. Handlin |  
Rappi Tech

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략

## FLY.IO

Kurt Mackey가 공동 설립한 회사로 개발. FLY.IO는 주로 개발자 친화적인 글로벌 애플리케이션 호스팅 서비스를 제공하기 위해 설립. Kurt는 이전 Compose.io라는 DB 호스팅 플랫폼을 공동 창업 경험(IBM에 매각)



회원가입 및 설치

FLY.IO 웹사이트(<https://fly.io>)에 접속하여 회원가입을 한다.

터미널에서 설치 스크립트 실행

```
pwsh -Command "iwr https://fly.io/install.ps1 -useb | iex"
```

터미널에서 로그인 실행

```
fly auth login
```

```
git clone https://github.com/fly-apps/hello-fly.git
cd hello-fly
```

```
fly launch --now
```

Daddy Makers: 가성비 있는 웹 서비스 호스팅 Fly.IO 사용하기

# AI 모델의 서비스화 ONNX, FastAPI, Flask 등 경량화 및 배포 전략

FLY.IO

Dashboard - Fly

https://fly.io/dashboard/taewook-kang

Apps

Search

Deployed Pending Suspended

- bim-data-quality-checker
- scan-to-model-app
- ai-agent-simple-function-call

Apps Team Billing Activity Usage Metrics Managed Postgres phoenix.new Upstash Redis Tigris Object Storage

Fly App - Dashboards - Grafana

https://fly-metrics.net/d/fly-app/fly-app?orgId=845238&var-app=...

Search or jump to... ctrl+k

Home > Dashboards > Fly App

Share Last 1 hour

Overview

Data Out

Edge

0 B/s 100 B/s

Instance

Network I/O

region	Instance
nrt	151 B/s

0 B/s  
-2 kB/s  
-4 kB/s  
-6 kB/s

11:00 11:10 11:20 11:30 11:40 11:50

Scan to Model Pipeline

https://scan-to-model-app.fly.dev

### Scan to Model Pipeline (ver 0.2. prototype)

Upload pipeline configuration file (JSON) and point cloud data file (LAS, LAZ) to process the data and download the results as a zip file.

1. [Upload pipeline configuration \(JSON\)](#)
2. [Upload point cloud data \(LAS, LAZ\)](#)
3. Click 'Run Pipeline' button

In detail, refer to the [github page](#)

Pipeline Configuration File

Drop File Here  
- OR -  
Click to Upload

Point Cloud Data File

Drop File Here  
- OR -  
Click to Upload

BIM Quality Checker

https://bim-data-quality-checker.fly.dev

### BIM Quality Checker (ver 0.46. prototype)

#### Instructions

1. Upload BIM Check Ruleset JSON Configuration File
2. Upload BIM Files (COBie.xlsx, csv, IFC, LandXML)
3. Click 'Run' to check BIM data quality
4. Download the Quality Report

Image

Upload BIM Check Ruleset JSON Configuration File

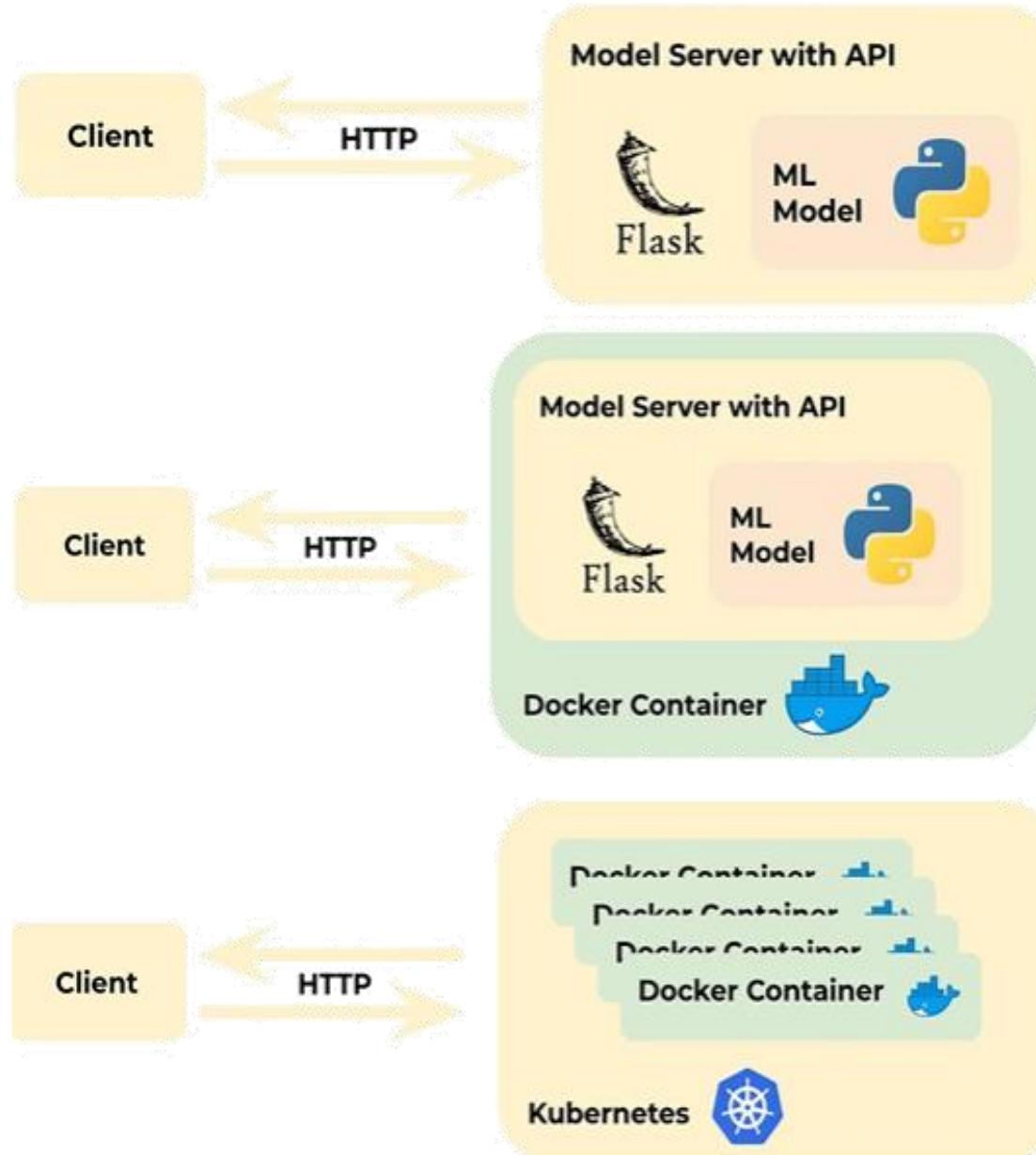
Upload BIM Files (COBie.xlsx, csv, IFC, LandXML)

BIM Viewer

Daddy Makers: 가성비 있는 웹 서비스 호스팅 Fly.IO 사용하기

# AI 모델의 서비스화

ONNX, FastAPI, Flask 등 경량화 및 배포 전략



Serve your first model with Scikit-Learn +  
Flask + Docker | by Carl W. Handlin |  
Rappi Tech

# AX전략

기업 AI 도입 시 고려사항  
질의응답 및 추천 학습 자료 안내

# AX 전략

## 주요 5개 실패 원인

Concerns about data accuracy or bias (45%)

Insufficient proprietary data available to customize models (42%)

Inadequate generative AI expertise (42%)

Inadequate financial justification or business case (42%)

Concerns about privacy or confidentiality of data and information (40%)



The image shows the top navigation bar of the IBM website. It features the IBM logo on the left, followed by a horizontal menu with dropdown arrows: AI, Hybrid Cloud, Products, Consulting, Support, and Think. Below this is a secondary navigation bar with links: Think (underlined in blue), Artificial intelligence, Cloud, X-Force, News, Videos, Reports (selected), Podcasts, Events, and More.

## The 5 biggest AI adoption challenges for 2025



# AX 전략

## 유스케이스

### Deloitte 보고서

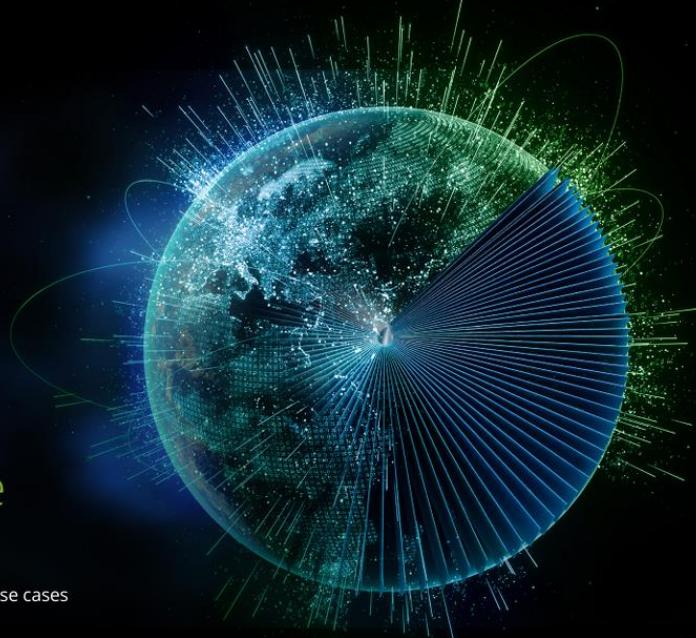
- 마케팅 컨텐츠 작성
- 코딩 개발 지원
- 고객 응대 지원
- 제품 설계 지원
- 연구 보고서 조사 작성
- 합성 및 학습 데이터 생성
- 기업 내 데이터 검색, 정리, 분석 및 요약
- 게임 컨텐츠 설계 및 개발
- 언어 번역
- 시뮬레이션 및 추론
- 고도화된 개인 맞춤 교육 및 훈련
- 현장 업무 지원

**Deloitte.**

## The Generative AI Dossier

A selection of high-impact use cases  
across six major industries

By Deloitte AI Institute



### About the Deloitte AI Institute

The Deloitte AI Institute™ helps organizations connect all the different dimensions of the robust, highly dynamic, and rapidly evolving Artificial Intelligence ecosystem. The AI Institute leads conversations on applied AI innovation across industries, with cutting-edge insights, to promote human-machine collaboration in the "Age of With".

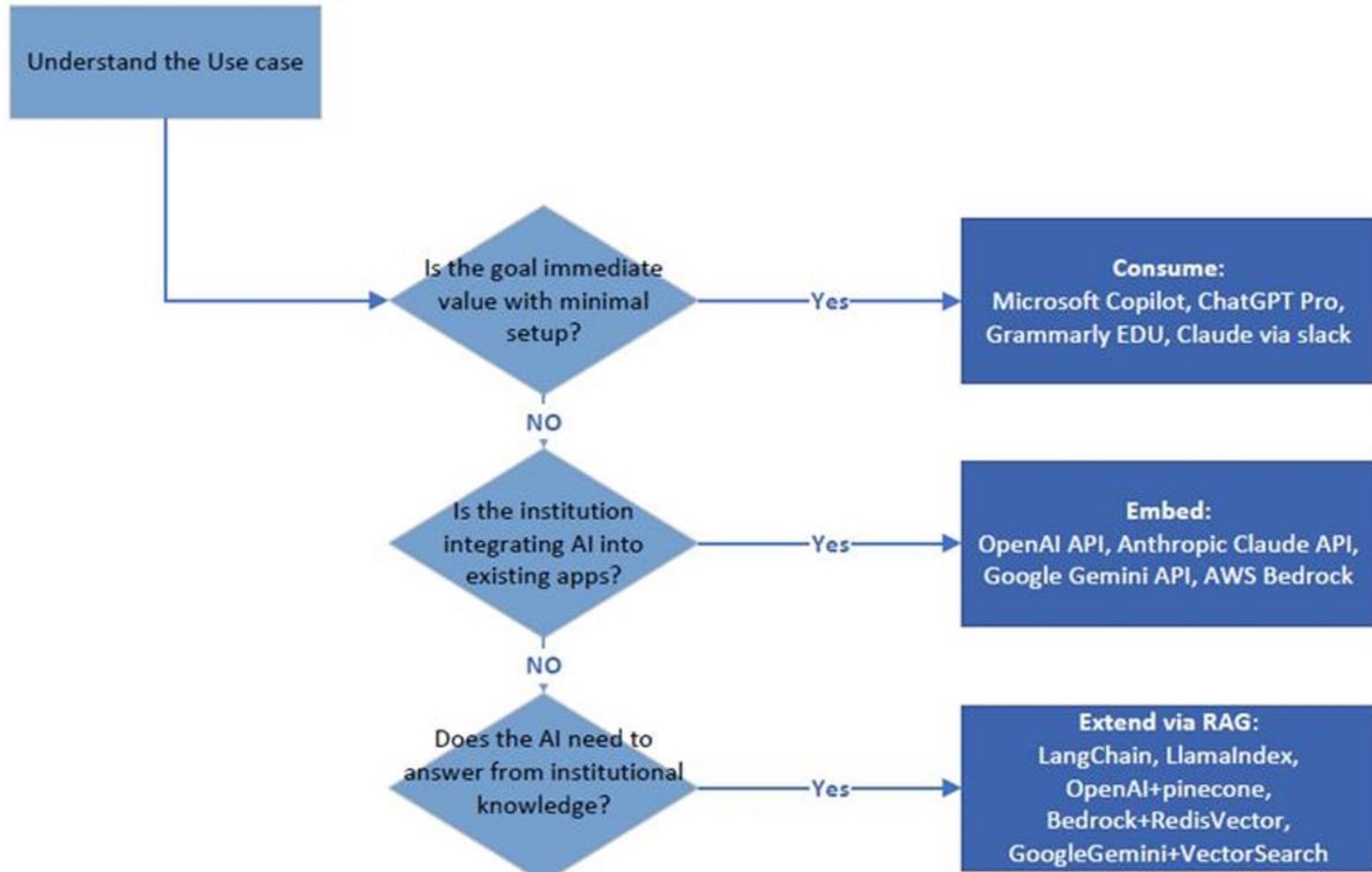
The Deloitte AI Institute aims to promote the dialogue and development of AI, stimulate innovation, and examine challenges to AI implementation and ways to address them. The AI Institute collaborates with an ecosystem composed of academic research groups, start-ups, entrepreneurs, innovators, mature AI product leaders, and AI visionaries to explore key areas of artificial intelligence including risks, policies, ethics, the future of work and talent, and applied AI use cases. Combined with Deloitte's deep knowledge and experience in artificial intelligence applications, the Institute helps make sense of this complex ecosystem, and as a result, delivers impactful perspectives to help organizations succeed by making informed AI decisions.

No matter what stage of the AI journey you are in: whether you are a board member or a C-Suite leader driving strategy for your organization—or a hands-on data scientist bringing an AI strategy to life—the Deloitte AI Institute can help you learn more about how enterprises across the world are leveraging AI for a competitive advantage. Visit us at the Deloitte AI Institute for a full body of our work, subscribe to our podcasts and newsletter, and join us at our meet-ups and live events. Let's explore the future of AI together.

[www.deloitte.com/us/AIinstitute](http://www.deloitte.com/us/AIinstitute)

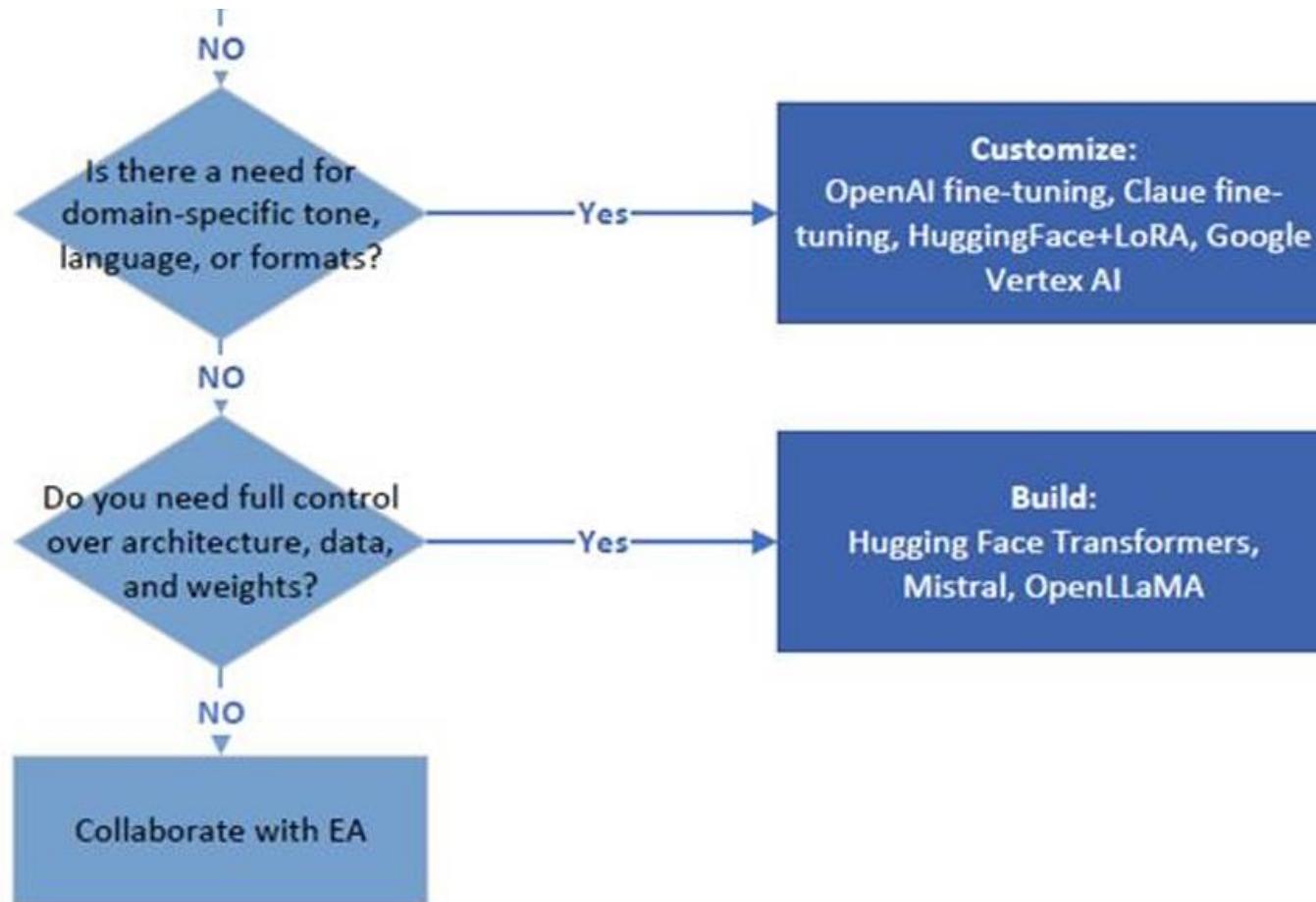
# AX 전략 기업 AI 도입 시 고려사항

## 유스케이스에 따른 AX 서비스 개발 의사결정 예시



# AX 전략 기업 AI 도입 시 고려사항

## 유스케이스에 따른 AX 서비스 개발 의사결정 예시



# AX 전략 기업 AI 도입 시 고려사항

## AI 도구 평가 방법 예시

### Appendix: AI Toolset Evaluation

To ensure objective and consistent selection of AI tools, Columbia proposes a scoring framework based on six key criteria. Each tool is scored on a scale of 1 to 10 using standardized rubrics, and then weighted to reflect institutional priorities.

#### Evaluation Criteria

Criterion	What It Measures
Performance	Speed, scalability, and accuracy under real-world workloads
Cost	Licensing, usage, infrastructure cost, and transparency
Customization	Ability to fine-tune or configure for domain-specific use
Compliance & Security	Support for regulatory/security standards (FERPA, HIPAA, etc.)
Integration & Tooling	Compatibility with systems, APIs, SDKs, and documentation
Community & Support	Docs, vendor responsiveness, and community activity

Criterion	Weight
Performance	25%
Cost	20%
Customization	15%
Compliance & Security	15%
Integration & Tooling	15%
Community & Support	10%

# AX 전략 기업 AI 도입 시 고려사항

## 오픈소스 도구

Search or jump to... Pull requests Issues Marketplace Explore

Overview Repositories 6 Stars 1 Followers 0 Following 0

Pinned repositories

- Automation: Rhino addin tool for generating tool path such as CNC, 3D printer etc. (C#)
- Books: CSS
- DeepLearning: Forked from tensorflow/models (Python)
- Projects: VHDL
- Robot: C++

Customize your pinned repositories

25 contributions in the last year

Contribution settings ▾

Jul Aug Sep Oct Nov Dec Jan Feb Mar Apr May Jun Jul

Tae wook  
mac999  
Ph.D. Senior researcher in KICT. Writer about 11 books including two e-books such as BIM principle, Civil BIM, platform, vision, reverse engineering.

Edit bio

KICT Seoul laputa9999@gmail.com https://sites.google.com/site/bi...

Search or jump to... Pull requests

mac999 / Automation

Code Issues 0 Pull requests 0 Projects 1

Branch: master Automation / OpenSlicer /

mac999 Delete OpenSlicer.v12.suo

..

Properties

bin

obj

AddSurfacePoints.cs

DelPoints.cs

GetSurfacePoints.cs

OpenSlicer.csproj

OpenSlicer.csproj.user

OpenSlicer.sln

OpenSlicer.suo

OpenSlicerBrep.cs

OpenSlicerPlugIn.cs

# AX 전략 기업 AI 도입 시 고려사항

## 오픈소스 도구

### GNU General Public License(GPL) 2.0

- 의무 엄격. SW 수정 및 링크 경우 소스코드 제공 의무

### GNU Lesser GPL(LGPL) 2.1

- 저작권 표시. LPGL 명시. 수정한 라이브러리 소스코드 공개

### Berkeley Software Distribution(BSD) License

- 소스코드 공개의무 없음. 상용 SW 무제한 사용 가능

### Apache License

- BSD와 유사. 소스코드 공개의무 없음.

### Mozilla Public License(MPL)

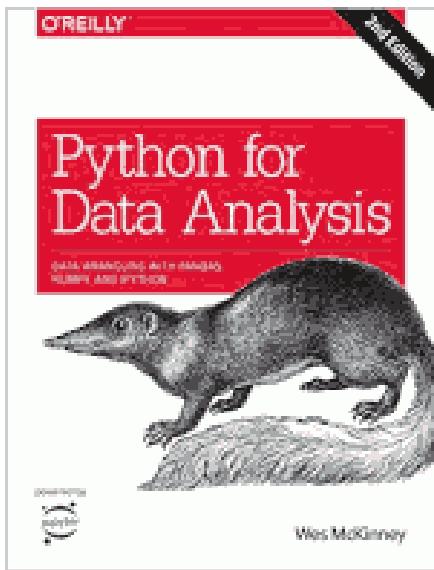
- 소스코드 공개의무 없음. 수정 코드는 MPL에 의해 배포. 이외 결합 프로그램 코드는 공개 필요 없음

### MIT License

- 라이선스 / 저작권만 명시 조건

# AX 전략과 트랜드

## 질의응답 및 추천 학습 자료 안내



OREILLY  
**NUMFOCUS**  
OPEN CODE = BETTER SCIENCE



VOLTRON DATA



Quansight Labs

TIDELIFT

bodo.ai

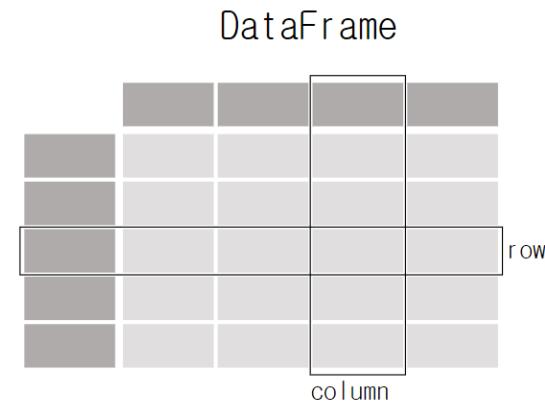
2 $\sigma$  TWO SIGMA

d-fine



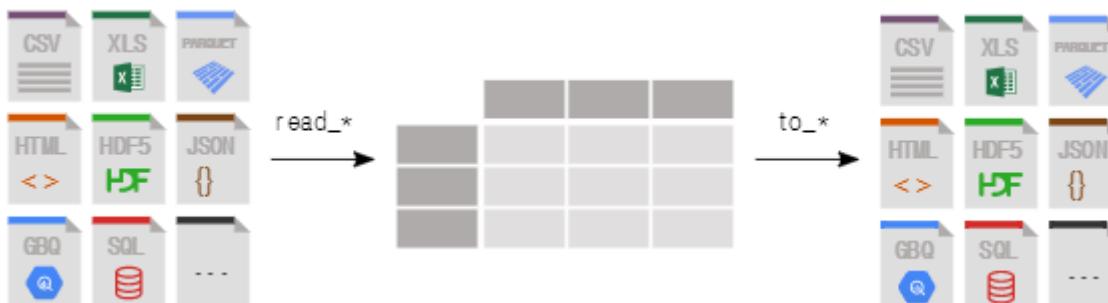
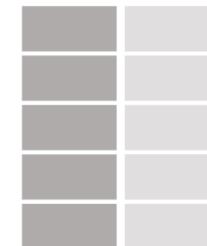
NVIDIA®

Chan  
Zuckerberg  
Initiative



Each column in a **DataFrame** is a **Series**

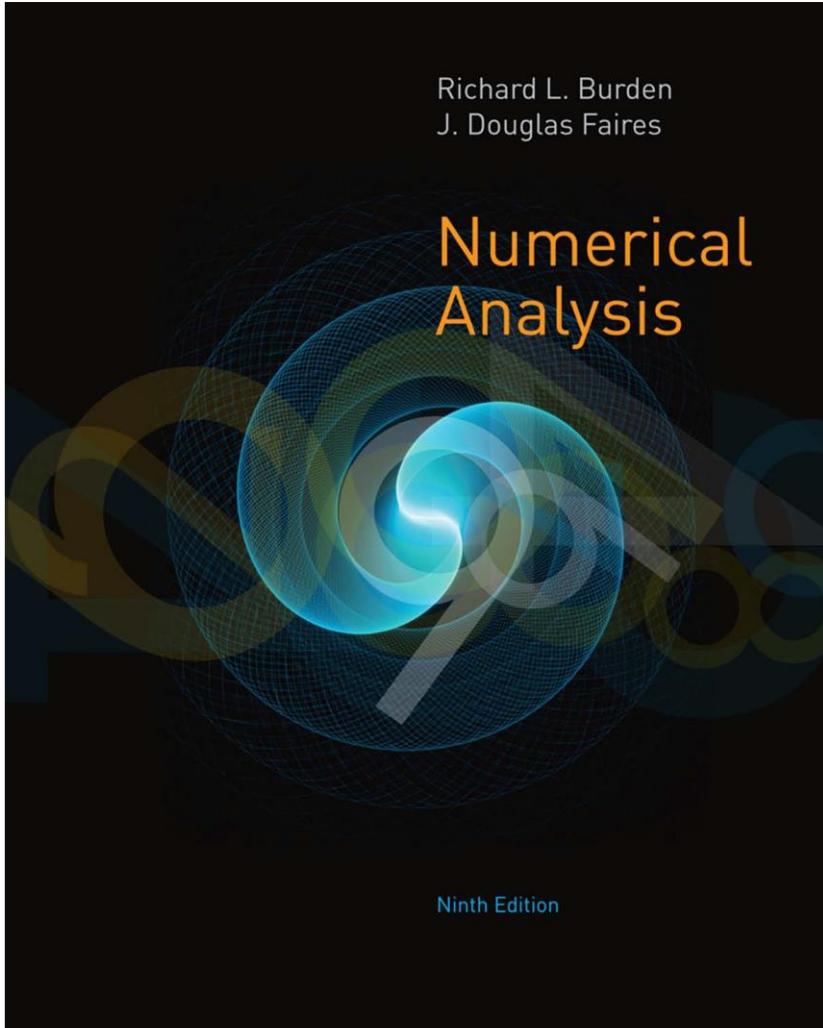
Series



<https://pandas.pydata.org/>

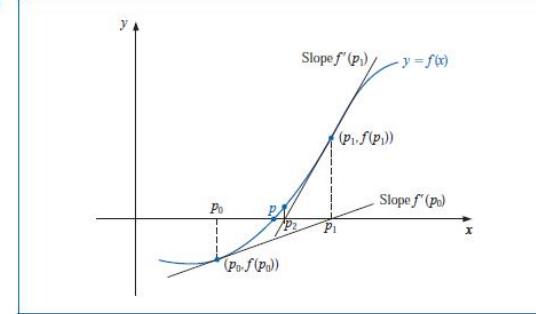
# AX 전략과 트랜드 질의응답 및 추천 학습 자료 안내

## 참고 자료



68 CHAPTER 2 ■ Solutions of Equations in One Variable

Figure 2.8



### Newton's

To find a solution to  $f(x) = 0$  given an initial approximation  $p_0$ :

**INPUT** initial approximation  $p_0$ ; tolerance  $TOL$ ; maximum number of iterations  $N_0$ .

**OUTPUT** approximate solution  $p$  or message of failure.

**Step 1** Set  $i = 1$ .

**Step 2** While  $i \leq N_0$  do Steps 3–6.

**Step 3** Set  $p = p_0 - f(p_0)/f'(p_0)$ . (Compute  $p_i$ .)

**Step 4** If  $|p - p_0| < TOL$  then  
    **OUTPUT** ( $p$ ); (The procedure was successful.)  
    STOP.

**Step 5** Set  $i = i + 1$ .

**Step 6** Set  $p_0 = p$ . (Update  $p_0$ .)

**Step 7** OUTPUT ('The method failed after  $N_0$  iterations,  $N_0 = ', N_0);  
(The procedure was unsuccessful.)  
STOP.$

The stopping-technique inequalities given with the Bisection method are applicable to Newton's method. That is, select a tolerance  $\varepsilon > 0$ , and construct  $p_1, \dots, p_N$  until

$$|p_N - p_{N-1}| < \varepsilon, \quad (2.8)$$

$$\frac{|p_N - p_{N-1}|}{|p_N|} < \varepsilon, \quad p_N \neq 0, \quad (2.9)$$

or

$$|f(p_N)| < \varepsilon. \quad (2.10)$$

# AX 전략과 트랜드 질의응답 및 추천 학습 자료 안내

## 참고 자료



### 목차(Contents)

<http://youtu.be/Mxple2Zzg-A>

Chapter 1. 벡터(Vectors)	*1.1 공학과 수학에서의 벡터: $n$ -공간 *1.2 내적과 직교 1.3 직선과 평면의 벡터방정식 연습문제
Chapter 2. 선형연립방정식	2.1 선형연립방정식 2.2 Gauss 소거법과 Gauss-Jordan 소거법 연습문제
Chapter 3. 행렬과 행렬대수	3.1 행렬연산 3.2 역행렬 3.3 기본행렬 3.4 부분공간과 일차독립 3.5 선형연립방정식의 해집합과 행렬 3.6 특수행렬들(Special matrices) 연습문제
Chapter 4. 행렬식	4.1 행렬식의 정의와 기본정리 4.2 여인자 전개와 행렬식의 응용 4.3 크래머 공식 *4.4 행렬식의 응용 4.5 고유값과 고유벡터
Chapter 5. 행렬모델	*5.1 Blackout Game *5.2 Sage를 활용한 선형모델 *5.3 퀴즈 및 중간고사 예시 <a href="http://matrix.skku.ac.kr/2014-Album/MC.html">http://matrix.skku.ac.kr/2014-Album/MC.html</a> <a href="http://matrix.skku.ac.kr/CLA-Exams-Sol.pdf">http://matrix.skku.ac.kr/CLA-Exams-Sol.pdf</a>
Chapter 6. 선형변환	6.1 함수(변환)로서의 행렬 6.2 선형변환의 기하학적 의미 6.3 핵과 치역 6.4 선형변환의 합성과 가역성 *6.5 Sage를 활용한 컴퓨터 그래픽 연습문제
Chapter 7. 차원과 부분공간	7.1 기저와 차원의 성질 7.2 행렬이 갖는 기본공간들 7.3 차원정리(Rank-Nullity 정리) 7.4 Rank 정리

# AX 전략과 트랜드

## 질의응답 및 추천 학습 자료 안내

### 참고 자료



CS231n: Deep Learning for  
Computer Vision

Lecture 1: Introduction

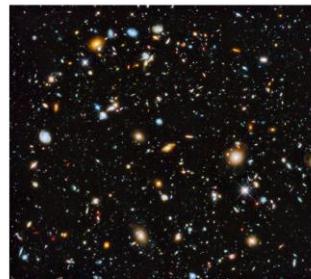
Fei-Fei Li & Ehsan Adeli

CS231n: Lecture 1 - 1

April 2, 2024



Lecture 1: Overview



CS221 / Spring 2019 / Charikar & Sadigh



deeplearning.ai

Introduction to  
Deep Learning

Welcome

AndrewNg

Brief Introduction to Natural  
Language Processing

EE599 Deep Learning

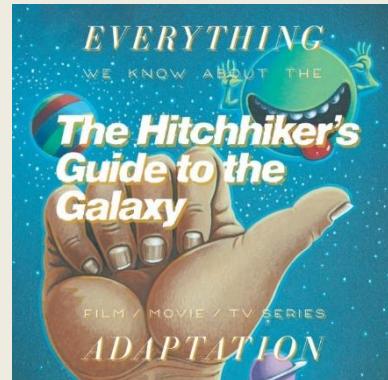
Keith M. Chugg  
Spring 2020



USC University of  
Southern California

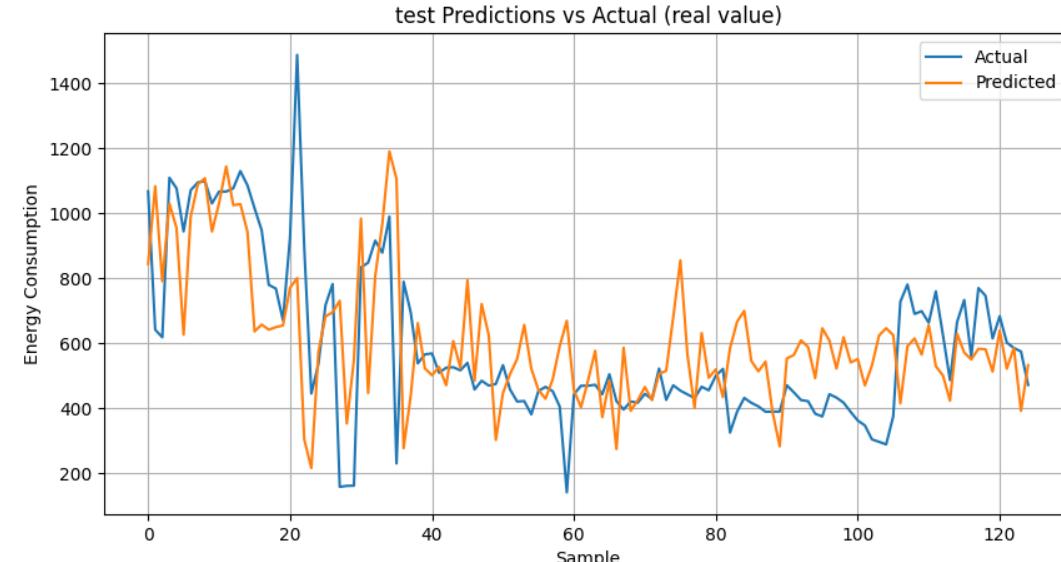
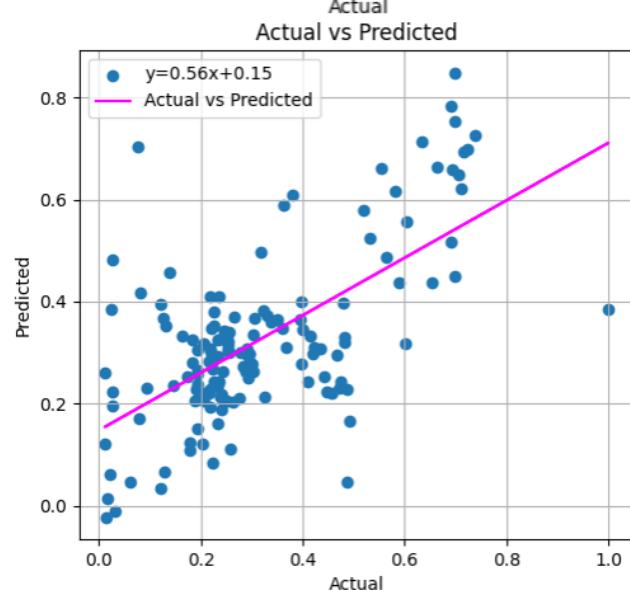
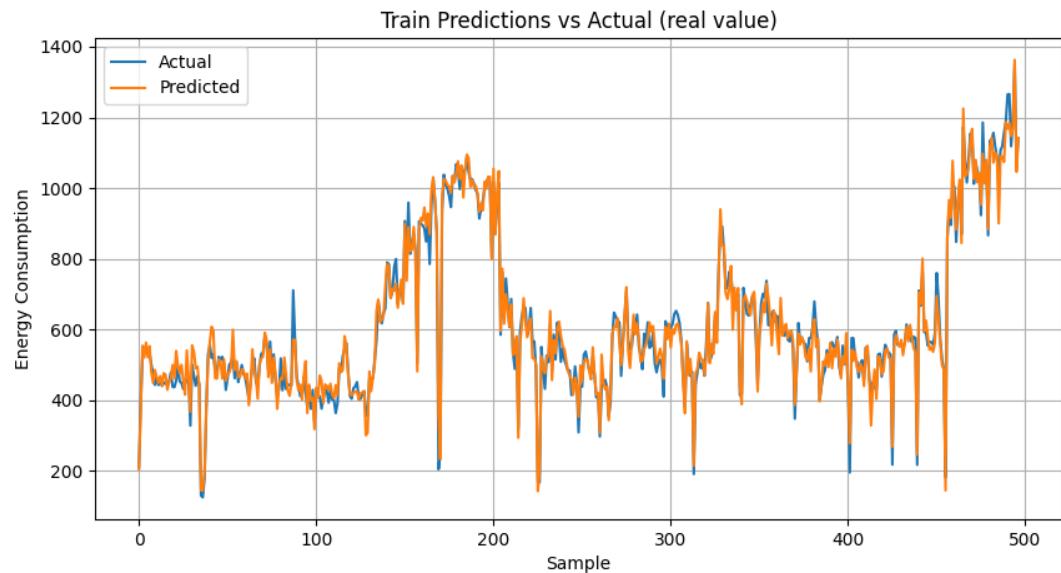
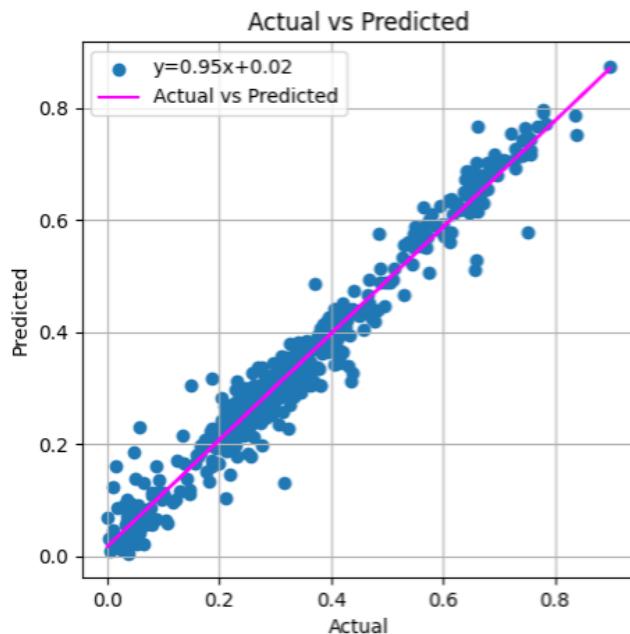
# Thanks

AX foundation and Trend 2025



laputa99999@gmail.com

# 개발 사례 에너지 소비 예측



Loss=0.002. Train MAPE & Acc = (0.025, 95.30%). Test MAPE & Acc = (0.292, 70.76%)

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

```
from math import exp  
from random import seed  
from random import random
```

## # Initialize a network

```
def initialize_network(n_inputs, n_hidden, n_outputs):  
    network = list()  
    hidden_layer = [{'weights':[random() for i in range(n_inputs + 1)]} for i in range(n_hidden)]  
    network.append(hidden_layer)  
    output_layer = [{'weights':[random() for i in range(n_hidden + 1)]} for i in range(n_outputs)]  
    network.append(output_layer)  
    return network
```

## # Calculate neuron activation for an input

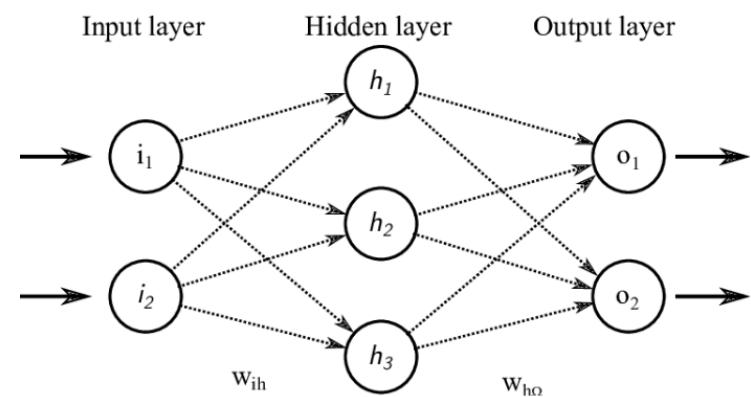
```
def activate(weights, inputs):  
    activation = weights[-1]  
    for i in range(len(weights)-1):  
        activation += weights[i] * inputs[i]  
    return activation
```

## # Transfer neuron activation

```
def transfer(activation):  
    return 1.0 / (1.0 + exp(-activation))
```

## # Forward propagate input to a network output

```
def forward_propagate(network, row):  
    inputs = row  
    for layer in network:  
        new_inputs = []  
        for neuron in layer:  
            activation = activate(neuron['weights'], inputs)  
            neuron['output'] = transfer(activation)  
            new_inputs.append(neuron['output'])  
        inputs = new_inputs  
    return inputs
```



INPUT = 2  
HIDDEN LAYERS  
OUTPUT = 2

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

```
# Calculate the derivative of an neuron output
def transfer_derivative(output):
    return output * (1.0 - output)           $\sigma(x)' = \sigma(x)(1 - \sigma(x))$        $\sigma(x) = 1/(1 + e^{-x})$ 

# Backpropagate error and store in neurons
def backward_propagate_error(network, expected):
    for i in reversed(range(len(network))):
        layer = network[i]
        errors = list()
        if i != len(network)-1:
            for j in range(len(layer)):
                error = 0.0
                for neuron in network[i + 1]:
                    error += (neuron['weights'][j] * neuron['delta'])
                errors.append(error)
        else:
            for j in range(len(layer)):
                neuron = layer[j]
                errors.append(neuron['output'] - expected[j])
        for j in range(len(layer)):
            neuron = layer[j]
            neuron['delta'] = errors[j] * transfer_derivative(neuron['output'])

# Update network weights with error
def update_weights(network, row, l_rate):
    for i in range(len(network)):
        inputs = row[:-1]
        if i != 0:
            inputs = [neuron['output'] for neuron in network[i - 1]]
        for neuron in network[i]:
            for j in range(len(inputs)):
                neuron['weights'][j] -= l_rate * neuron['delta'] * inputs[j]
            neuron['weights'][-1] -= l_rate * neuron['delta']
```

# 딥러닝의 구조적 이해

Optimizer, Learning Rate, Batch Size 의미

```
# Train a network for a fixed number of epochs
def train_network(network, train, l_rate, n_epoch, n_outputs):
    for epoch in range(n_epoch):
        sum_error = 0
        for row in train:
            outputs = forward_propagate(network, row)
            expected = [0 for i in range(n_outputs)]
            expected[row[-1]] = 1
            sum_error += sum([(expected[i]-outputs[i])**2 for i in range(len(expected))])
            backward_propagate_error(network, expected)
            update_weights(network, row, l_rate)
        print('>epoch=%d, lrate=%f, error=%f' % (epoch, l_rate, sum_error))

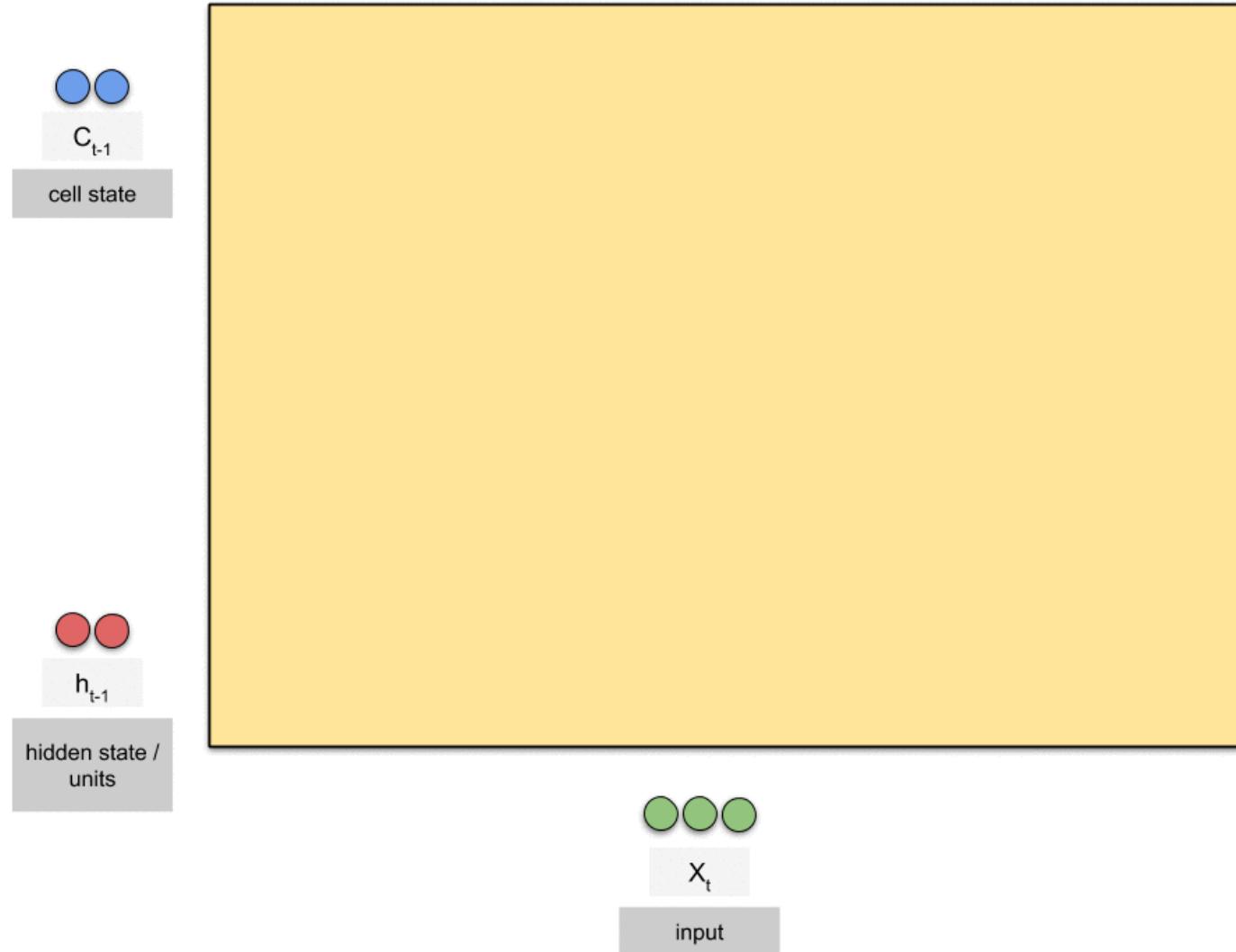
# Test training backprop algorithm
seed(1)
dataset = [[2.7810836,2.550537003,0],
           [1.465489372,2.362125076,0],
           [3.396561688,4.400293529,0],
           [1.38807019,1.850220317,0],
           [3.06407232,3.005305973,0],
           [7.627531214,2.759262235,1],
           [5.332441248,2.088626775,1],
           [6.922596716,1.77106367,1],
           [8.675418651,-0.242068655,1],
           [7.673756466,3.508563011,1]]
n_inputs = len(dataset[0]) - 1
n_outputs = len(set([row[-1] for row in dataset]))
network = initialize_network(n_inputs, 2, n_outputs)
train_network(network, dataset, 0.5, 20, n_outputs)
for layer in network:
    print(layer)
```

>epoch=0, lrate=0.500, error=6.350  
>epoch=1, lrate=0.500, error=5.531  
>epoch=2, lrate=0.500, error=5.221  
>epoch=3, lrate=0.500, error=4.951  
>epoch=4, lrate=0.500, error=4.519  
>epoch=5, lrate=0.500, error=4.173  
>epoch=6, lrate=0.500, error=3.835  
>epoch=7, lrate=0.500, error=3.506  
>epoch=8, lrate=0.500, error=3.192  
>epoch=9, lrate=0.500, error=2.898  
>epoch=10, lrate=0.500, error=2.626  
>epoch=11, lrate=0.500, error=2.377  
>epoch=12, lrate=0.500, error=2.153  
>epoch=13, lrate=0.500, error=1.953  
>epoch=14, lrate=0.500, error=1.774  
>epoch=15, lrate=0.500, error=1.614  
>epoch=16, lrate=0.500, error=1.472  
>epoch=17, lrate=0.500, error=1.346  
>epoch=18, lrate=0.500, error=1.233  
>epoch=19, lrate=0.500, error=1.132  
[{'weights': [-1.468837509543237, 1.850887325439514, 1.0858178629550297], 'output': 0.029983625}, {'weights': [0.37711098142462157, -0.0625909894552989, 0.2765123702642716], 'output': 52850863837}]  
[{'weights': [2.515394649397849, -0.3391927502445985, -0.9671565426390275], 'output': 0.236487}, {'weights': [-2.5584149848484263, 1.0036422106209202, 0.42383086467582715], 'output': 6437354}]

# 딥러닝의 구조적 이해

CNN/RNN/LSTM의 개념적 차이

## LSTM (Long Short Term Memory)



[Animated RNN, LSTM and GRU.](#)  
[Recurrent neural network cells in GIFs](#)  
[by Raimi Karim]  
[TDS Archive]  
Medium