

현업의 딥러닝 개발자가 말하는 딥러닝 개발의 현황과 미래

(주)포지큐브 기술연구소 김건호 AI기술이사

강사 약력

기간	소속	주요 업무	직급 또는 직책
~2002년 2월	▪ 한국해양대학교 대학원	▪ 응용수학(암호학)	▪ 석사
~2005년 5월	▪ (주)애드시큐	▪ 암호시스템 개발 및 관리	▪ 개발팀 대리
~2010년 3월	▪ 에디슨아카데미	▪ 수학강의 ▪ 학생 및 강사 관리	▪ 부원장/강사
~2017년 11월	▪ (주)큐램(기술연구소)	▪ 영상압축 기술 개발 ▪ iOS/AOS APPLICATION 개발 및 최적화 ▪ Engine Library 최적화 기술 개발 ▪ CV 활용 다양한 기술 개발 ▪ ML/DL 활용 인식 기술 개발 ▪ 개발 관리	▪ 수석연구원/영상처리 팀장
~2021년 5월	▪ (주)아모레퍼시픽(본사)	▪ CV 활용 기술 개발 ▪ Image/Video Filter 기술 개발 ▪ ML/DL 활용 분석 기술 개발 ▪ 개발 관리	▪ AI 개발 리더/스마트뷰티 파트장
~현재(2021년 12월)	▪ (주)포지큐브(기술연구소)	▪ CV 활용 기술 개발 ▪ ML/DL 활용 검출/인식 기술 개발 ▪ 개발 관리	▪ AI 기술 이사/VisionAI 팀장

강의 계획서

강의 제목	현업의 딥러닝 개발자가 말하는 딥러닝 개발의 현황과 미래		
강사 정보	<ul style="list-style-type: none"> 강사명 : 김건호 이메일 : niki@posicube.com 연락처 : 010-2922-2076 	강의 개요	<ul style="list-style-type: none"> 강의 목표 : <ul style="list-style-type: none"> 딥러닝의 개요 및 현업에서의 적용분야와 연관업무에 대한 이해 딥러닝 개발 환경 설정 연습 영상분야의 딥러닝 개발 예제를 통한 실사용 연습 실제 딥러닝 적용 한계와 미래에 대한 고찰 학생들에 대한 지도 방향 설정 강의 내용 <ul style="list-style-type: none"> 딥러닝 개요, 딥러닝 적용분야 개요, 딥러닝 연관업무 소개 딥러닝의 종류, 딥러닝 기술 개발을 위한 준비 영상분야의 딥러닝을 위한 사전 지식 딥러닝 실습(이미지 인식 MNIST 학습 등) 현업의 딥러닝 실제 적용 분야와 한계, 딥러닝의 미래 딥러닝 업무를 위한 학생 지도 방안 Q&A
강의 시간	<ul style="list-style-type: none"> 이론 : 1시간~1시간30분 실습 : 2시간~2시간30분 Q&A : 30분 <p>총 : 3시간30분~4시간30분</p>		
강의 대상	<ul style="list-style-type: none"> 대구소프트웨어마이스트고등학교 교사(약 10명) 		
준비 사항	<ul style="list-style-type: none"> 강의 자료 : PDF 사전제공 실습 준비물 : CPU i7급 이상 PC 또는 노트북 (GTX 1060급 이상 GPU 장착 추천) 		

회사 소개

회사명

(주)포지큐브

대표자

오성조(삼성전자 출신)

주소

서울특별시 강남구 테헤란로 14길 16 라인빌딩 11층

주요 솔루션

로비 리셉션 : 통합 상담 AI 음성 챗봇 솔루션
로비 스캐너 : 신용카드, 신분증 OCR
문서 OCR : 각종 공공 행정 문서, 금융 문서 OCR

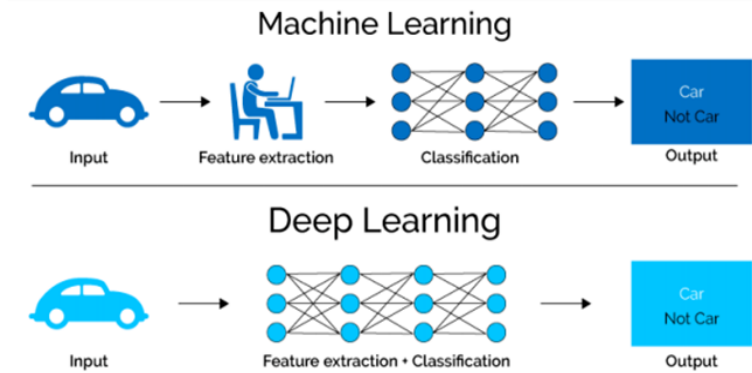
회사 소개글

포지큐브는 AI 기술 전문 회사로,
공공기관 및 민간기업에게 AI Voice & Vision 서비스를 제공하고 있습니다.**
우리는 뉴스에서 들어본 것 같은 AI 기술이 아닌, 삶에 적용되는 AI 서비스를 제공하고 있습니다. AI 기술이 우리 삶의 당연한 일부분이 되는 그 날을 향해 노력하고 있습니다. AI 전화 응대 서비스, AI based OCR 서비스 등의 AI 서비스를 개발·운영합니다.

1.1 딥러닝 개요(개념)



그림4 머신러닝 VS 딥러닝



자료: Towards Data Science, 메리츠증권증권 리서치센터

- **인공지능(AI, Artificial Intelligence)**

인공지능은 인간의 지능이 갖고 있는 기능을 갖춘 컴퓨터 시스템이며, 인간의 지능을 기계 등에 인공적으로 시연(구현)한 가장 큰 범주에 해당합니다. 일반적으로 범용 컴퓨터에 적용한다고 가정합니다.

- **머신러닝(Machine Learning)**

기계 학습 또는 머신 러닝은 인공지능의 한 분야로, 컴퓨터가 학습할 수 있도록 하는 알고리즘과 기술을 개발하는 분야를 말합니다.

- **딥 러닝(Deep Learning)**

심층 학습 또는 딥러닝은 여러 비선형 변환기법의 조합을 통해 높은 수준의 추상화를 시도하는 기계 학습 알고리즘의 집합으로 정의 됩니다. 큰 틀에서 사람의 사고방식을 컴퓨터에게 가르치는 기계학습의 한 분야라고 이야기할 수 있습니다. 딥러닝은 특징 추출부터 패턴까지 모든 과정을 사람의 개입 없이 심층인공신경망을 토대로 학습방식을 구현하는 기술입니다.

머신러닝은 인간이 데이터에 대한 결과 값을 미리 알려주어야 하고 목표치에 가까운 결과 값의 특징을 미리 정의해야 합니다.

반면에 딥러닝은 인간의 신경망인 뉴런의 작동 원리를 모방한 인공 신경망을 이용하여 복잡하고 방대한 데이터로부터 결과값을 추출하는 원리로 작동합니다.

1.2 딥러닝 개요(학습 방식)

- **지도학습(Supervised learning)**

지도학습에는 기존에 이미 분류된 학습용 데이터(labeled trainig data)로 구성된 입력 변수와 원하는 출력 변수가 수반됩니다. 예를 들어, 과거 매출 이력을 이용해 미래 가격을 추산할 수 있습니다. 알고리즘을 이용해 학습용 데이터를 분석함으로써 입력 변수를 출력 변수와 매핑시키는 함수를 찾을 수 있습니다. 이렇게 추론된 함수는 학습용 데이터로부터 일반화(generalizing)를 통해 알려지지 않은 새로운 사례들을 매핑하고, 눈에 보이지 않는 상황 속에서 결과를 예측합니다. 지도 학습은 학습 결과를 바탕으로 미래의 무엇을 예측하냐에 따라 분류, 회기, 예측으로 구분할 수 있습니다.

- **비지도학습(Unsupervised learning)**

비지도 학습은 수행할 때 기계는 미분류 데이터만을 제공 받습니다. 그리고 기계는 클러스터링 구조(clustering structure), 저차원 다양체(low-dimensional manifold), 희소 트리 및 그래프(a sparse tree and graph) 등과 같은 데이터의 기저를 이루는 고유 패턴을 발견하도록 설정됩니다.

- **강화학습(Reinforcement learning)**

강화 학습은 환경으로부터의 피드백을 기반으로 행위자(agent)의 행동을 분석하고 최적화 합니다. 기계는 어떤 액션을 취해야 할지 듣기 보다는 최고의 보상을 산출하는 액션을 발견하기 위해 서로 다른 시나리오를 시도합니다. 시행 착오(Trial-and-error)와 지연 보상(delayed reward)은 다른 기법과 구별되는 강화 학습만의 특징입니다.



1.3 딥러닝 개요(학습 목적)

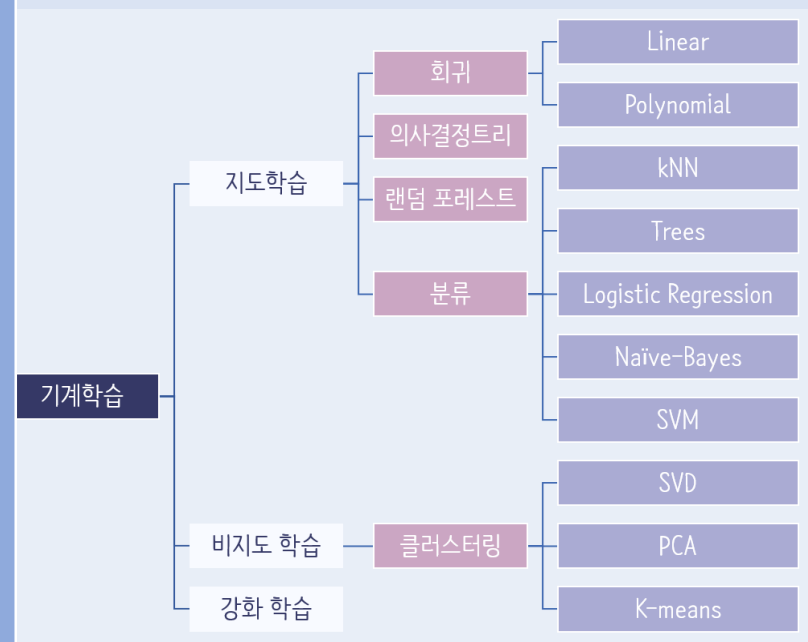
지도학습(Supervised learning)

- **분류(Classification):**
데이터가 범주형 변수를 예측하기 위해 사용될 때 지도학습을 '분류'라고 하기도 합니다. 이미지에 강아지나 고양이와 같은 레이블을 할당하는 경우가 해당됩니다. 레이블이 두 개인 경우를 이진 분류(binary classification)라고 부르며, 범주가 두 개 이상인 경우는 다중 클래스 분류(multi-class classification)라고 부릅니다.
- **회귀(Regression):**
연속 값을 예측할 때 문제는 회귀 문제가 됩니다. 트레이닝 데이터를 이용하여 연속적인 값을 예측하는 것을 말합니다.
- **예측(Forecasting):**
과거 및 현재 데이터를 기반으로 미래를 예측하는 과정입니다. 예측은 동향(trends)을 분석하기 위해 가장 많이 사용됩니다. 예를 들어 올해와 전년도 매출을 기반으로 내년도 매출을 추산하는 과정입니다.

비지도학습(Unsupervised learning)

- **클러스터링(Clustering):**
특정 기준에 따라 유사한 데이터 사례들을 하나의 세트로 그룹화합니다. 이 과정은 종종 전체 데이터 세트를 여러 글부로 분류하기 위해 사용됩니다. 사용자는 고유한 패턴을 찾기 위해 개별 그룹 차원에서 분석을 수행할 수 있습니다.
- **차원 축소(Dimension Reduction):**
고려 중인 변수의 개수를 줄이는 작업입니다. 많은 애플리케이션에서 원시 데이터(raw data)는 아주 높은 차원의 특징을 지닙니다. 이때 일부 특징들은 중복되거나 작업과 아무 관련이 없습니다. 따라서 차원수를 줄이면 관계를 도출하기 용이해집니다.

강화학습(Reinforcement learning)

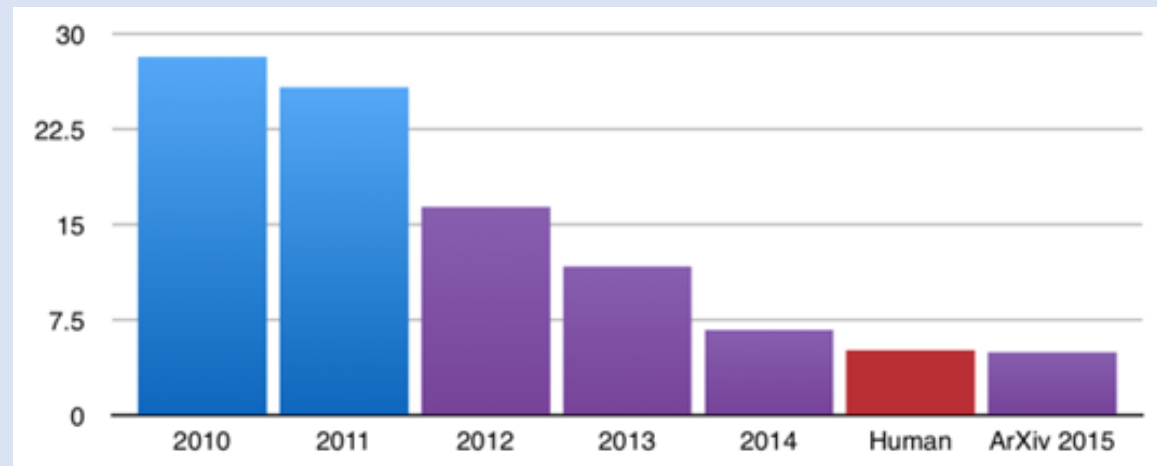


2.1.1 딥러닝 적용분야(개요)

컴퓨터 비전 (Computer Vision)

- 이미지 분류 (Image classification)

인간이 이미지를 분류하는 성능은 약 95%정도 입니다. 인공지능 모델을 구축해서 이미지를 분류하는 대회가 2010년부터 열려왔습니다. 2010년에 우승했던 모델의 분류 성능은 약 72%입니다. 어떻게 하면 인간처럼 또는 그 이상 이미지 분류의 성능을 낼 수 있을까 많은 연구자들이 고민을 해왔습니다. 2015년 ResNet이라는 모델이 약96%의 성능을 기록하면서 나오면서 인간의 성능을 뛰어넘기 시작했습니다. 그 이후로도, 딥러닝 모델들은 계속 발전해 오고 있습니다. 이제는 단순히 이미지를 분류하는 것을 넘어서 다양한 분야, 다양한 방식으로 발전해 오고 있습니다. 이미지 분류를 하더라도 수많은 데이터 없이 적은 데이터만을 가지고 어떻게 하면 높은 성능을 기록을 할 수 있을 지, 어떻게 하면 더욱 더 강건한 (Robust) 모델을 만들 수 있을지, 어떻게 하면 학습 데이터에 내에 있는 노이즈 데이터를 걸러 낼 수 있을지 등 다양한 형태와 분야로 발전해 오고 있습니다.

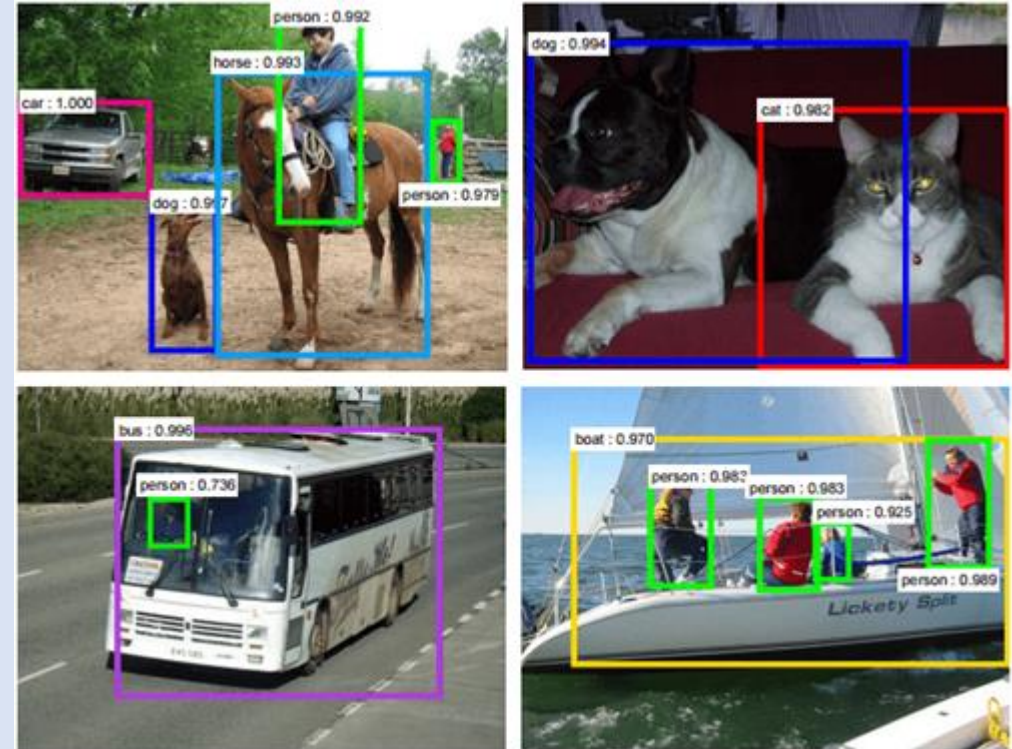


2.1.2 딥러닝 적용분야(개요)

컴퓨터 비전 (Computer Vision)

■ 객체 탐지 (Object Detection)

객체 탐지란 어떠한 이미지 및 비디오 속에 포함 되어있는 물체들에 대해서 어떤 물체인지 분류해내는 이미지 분류를 하는 것과 동시에 해당 물체가 이미지 및 비디오 속에 어디에 위치하였는지 찾아내는 일입니다. 다량의 이미지 및 비디오 데이터를 활용하며, 이미지 및 비디오 내 특정 물체의 위치 정보를 X, Y 좌표 값과 해당 물체의 크기인 Width, Height 값을 레이블 정보로 이용하여 딥러닝 모델이 학습합니다. 연구자들 사이에서는 (X, Y, W, H) 정보를 보통 Bounding Box라고 많이 표현하고 있습니다. 객체 탐지 기술은 자율주행자동차, CCTV 등 카메라 기술을 바탕으로 이루어지고 있는 제품 및 서비스에 최근 도입되고 있으며, 현재 다양한 연구가 진행되고 있습니다



2.1.3 딥러닝 적용분야(개요)

컴퓨터 비전 (Computer Vision)

▪ Segmentation

Segmentation은 앞서 Object Detection보다 정교한 탐지를 요구하는 연구분야입니다. 앞서 제시한 Object Detection 연구 분야는 특정 물체를 사전에 정의된 특정 클래스로 분류하며 입력값으로 이용된 이미지 혹은 비디오 화면 내 어느 위치에 존재하는지 파악하는 연구 분야입니다. 여기서, 어느 위치에 존재하는지 파악할 때, (x, y, w, h) 정보를 이용하여 Boundary Box를 그리게 되는데, 이 Boundary Box 내 물체가 꼭 차 있을 순 있지만, 해당 물체가 아닌 영역이 존재할 수 있습니다. Segmentation은 특정 위치에 Boundary Box로 물체 존재 유무로 표현하는 것의 한계를 극복하기 위해, 이미지 및 비디오 내 존재하는 모든 픽셀에 대해 특정 클래스로 예측하는 방식으로 진행합니다. 이미지 및 비디오를 표현하는 최소 단위인 픽셀 수준으로 접근하여, 각 픽셀별로 특정 클래스를 예측한다면, 예측된 정보를 바탕으로 픽셀에 색을 다르게 표현하여 이미지로 표현한다면, 이미지 및 비디오 내 모든 영역이 특정 클래스로 표현됩니다. 이해를 돕기 위해 그림을 하나 첨부하겠습니다.

특정 클래스로 분류된 물체에 대해 (x, y, w, h) 로 위치 정보를 표현하여 Boundary Box를 그린 Object Detection 분야는 그림을 통해 확인할 수 있듯이 사람이 아닌 다른 영역도 Boundary Box가 포함하고 있습니다. 하지만, 이미지 내 존재하는 모든 픽셀에 대해 클래스로 분류한 후 클래스 별로 색을 다르게 표현하여 이미지로 형상화 한 Semantic Segmentation을 보면, 사람의 위치에 해당하는 픽셀들은 분홍색, 사람이 아닌 위치의 픽셀들은 검정색으로 표현하여 이미지에 대해 물체의 클래스 및 위치 정보를 Object Detection에 비해 정확히 추출할 수 있는 장점을 갖고 있습니다.

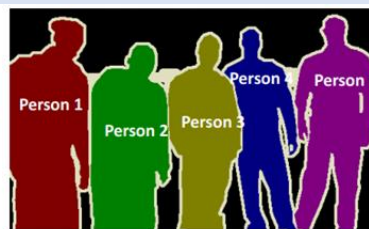
하지만, 동일한 클래스로 분류된 복수의 물체들의 픽셀값들이 바로 옆에 존재하여 연결되어 있는 경우, 여러 물체가 하나의 물체로 인식될 수 있는 문제가 존재합니다. 위의 그림에서 Semantic Segmentation 이미지를 살펴보면, Person이 5명의 사람이라고 우리는 바로 알아차릴 수 있지만, 컴퓨터 입장에서는 머리가 5개이고 다리가 7개인 사람 1명이라고 판단하게 됩니다. 따라서, 각 물체 별로 해당되는 픽셀값들을 구분하고, 구분된 픽셀값들에 대해서 클래스를 예측하는 Instance Segmentation 방식의 연구분야도 존재합니다.



Object Detection



Semantic Segmentation



Instance Segmentation

2.1.4 딥러닝 적용분야(개요)

자연어처리 (Text)

■ 텍스트 (Text)

텍스트 분야에서의 딥러닝 적용 또한 꾸준히 연구되었습니다. 텍스트 분야는 세부 Task로 나뉘어 연구가 되었는데 대표적으로 다음과 같은 것들이 있습니다.

이미지 영역에서는 딥러닝 모델이 인간보다 더 좋은 성능을 보이는 모습을 보인 반면, 텍스트 분야에서는 인간의 성능을 따라잡기 어려웠습니다. 이미지에 비해 텍스트 Task는 배경지식이 요구된다는 점도 어려운 점이었고, 주로 사용한 Recurrent Neural Network (RNN) 계열의 모델의 한계 역시 해결해야 할 문제 중 하나였습니다. 하지만 2017년 구글이 발표한 Attention Is All You Need 라는 논문의 Transformer 모듈 연구를 시작으로 이와 관련된 모델들이 활발히 연구되기 시작했고, 인간의 성능을 넘어서는 Language Model이 개발되기 시작했습니다. 그리고 최근에는 이를 이용하여 학계에서는 다양한 분야의 추가 연구가, 산업계에서는 이와 관련된 서비스나 제품 연구가 활발히 이루어지고 있습니다.

- 언어 번역 (Machine Translation)
- 문장(혹은 문서) 분류 (Sentence Classification)
- 질의 응답 시스템 (Question & Answer System : Q/A)
- 개체명 인식 (Named Entity Recognition : NER)

■ 음성(Voice)

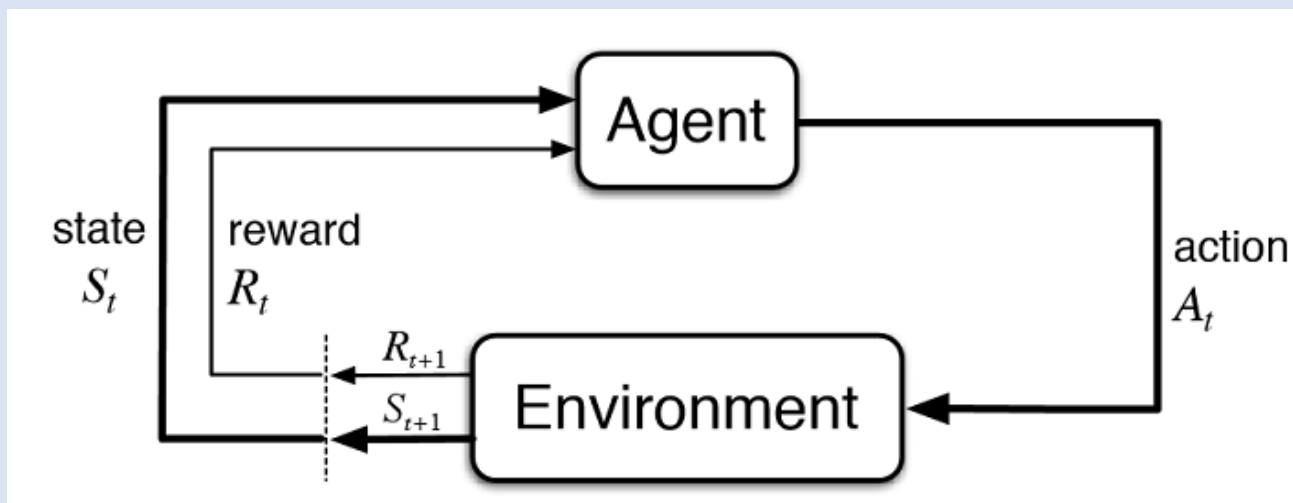
- TTS : 텍스트를 음성으로 변환하는 기술로 자연스러운 표현이 될 수 있도록 개발하는 것이 중요하다.
- STT : 음성을 텍스트로 변환하는 기술로 정확한 텍스트로 변환할 수 있도록 개발하는 것이 중요하다.

2.1.5 딥러닝 적용분야(개요)

강화학습 (Reinforcement Learning)

- 알파고 (AlphaGo)

딥러닝 또는 강화학습 (Reinforcement Learning)이라는 말은 들어보지 못 했어도 알파고라는 말은 들어봤을 겁니다. 알파고는 구글 (Google)의 딥마인드 (DeepMind)가 개발한 인공지능 바둑기사로 2016년 한국의 이세돌 기사와 대국해 4승 1패로 승리하며 세상을 놀라게 하였습니다. 이 알파고의 기본 원리는 강화학습으로 현재 상태 (바둑판)에서 어떠한 행동 (수)을 취해야 먼 미래에 보상이 최대 (승리)가 될 것인지에 대해서 학습하는 알고리즘입니다. 이는 수많은 시뮬레이션을 해야 하지만 가능합니다. 하지만 바둑의 경우의 수는 무한대에 가깝기 때문에, 인공지능이 바둑을 두기는 어렵다라는게 많은 전문가들의 의견이었습니다. 그러나 딥마인드는 이 강화학습 알고리즘과 딥러닝을 통해 문제를 해결하기에 이르렀습니다. 처음에는 학습 해야 할 경우의 수를 줄이기 위해 바둑의 기보를 학습하였으나, 2018년에는 알파고 제로 버전을 발표하면서 기보 없이 스스로 학습 해 나가는 인공지능 바둑기사를 개발 하였습니다. 알파고의 등장을 계기로 강화학습과 딥러닝을 결합한 심층 강화학습 (Deep Reinforcement Learning)의 연구가 활발히 진행되기 시작하였습니다.



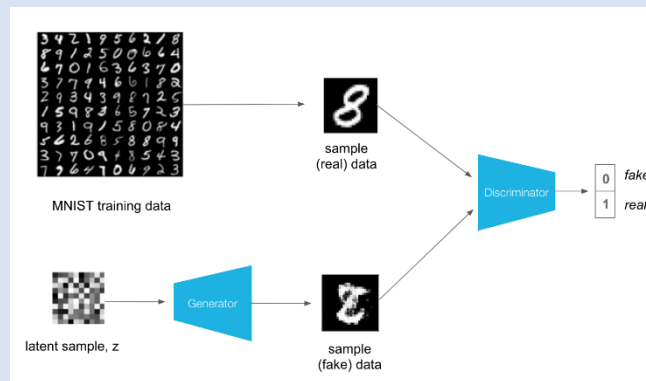
2.1.6 딥러닝 적용분야(개요)

Generative Adversarial Networks (GAN)

■ Generative Adversarial Networks (GAN)

딥러닝은 기본적으로 이미지와 텍스트 분류 쪽에 초점을 맞추어 발전해왔습니다. 그러면서 점점 더 다양한 분야, 다양한 방식으로 발전을 해왔습니다. 그러던 와중에 인공지능 분야에서 획기적인 일이 발생합니다. 바로 생성 모델 GAN의 등장인데요. 대중들에게 알파고가 엄청난 각인을 끼쳤다면 GAN은 연구자들에게 엄청난 각인을 끼쳤습니다. GAN은 2014년 Ian Goodfellow의 박사논문으로 처음 제안되었고, 우리나라에서 본격적으로 알려진 건 2016년 즈음입니다. 무언가 '예측'하는 걸 넘어서 '생성'해내는 모델을 제안을 한 것입니다. 이미지 분류 모델을 예로 들면 Input데이터가 이미지이고 딥러닝 모델의 Output은 Label입니다. 이 이미지가 고양이인지 강아지 인지 분류하는 모델인 것이죠. 그러나 이 GAN은 input이 Random Noise이고 Output이 임의의 이미지입니다. 학습하는 대상에 따라서 숫자 이미지가 될 수도 있고, 사람 이미지가 될 수도 있습니다. 즉, 내가 숫자 이미지를 만들어내는 딥러닝 모델을 만들고자 한다면, Input Noise, Output을 숫자 이미지로 하여 GAN을 학습하면 되는 것입니다.

딥러닝 모델로 분류 외에 생성을 한다는 것은 많은 이들에게 충격으로 다가왔습니다. GAN의 등장은 알파고의 등장과 비슷하게 인공지능 연구의 새로운 패러다임을 제시하였습니다. 처음에는 일부사람들은 생성되는 데이터의 질이 그렇게 좋지 않으므로 사용할 분야가 마땅치 않다고 지적하였으나, 최근에 연구되고있는 GAN의 성능은 이미 인간의 눈으로 구분하지 못할 정도의 고품질의 이미지 (또는 텍스트)를 생성해냅니다. 아래 그림은 2018년 발표 BigGAN (Large Scale GAN Training for High Fidelity Natural Image Synthesis, Brock et al., 2018) 이 생성한 이미지입니다. 즉, 이 세상에 실제로 존재하지 않는 객체에 대한 이미지라는 것이죠. 단순히 데이터를 생성해내는 것 뿐만 아니라, GAN이 가지는 학습 알고리즘의 특성을 이용하여 다양한 분야로 발전되어 오고 있습니다.



2.2 딥러닝 적용분야(현업)

현업의 딥러닝 적용 분야

■ 영상

- 얼굴인식 : 출입 통제, 인증, 신분증 확인
- 자동차 번호판 인식 : 주차 정산 자동화,
- 카드/신분증 인식 : 카드 번호/이름, 신분증 이름/주민번호 등을 인식하여 사용자 입력란에 자동 입력
- 문서인식 : 문서의 전체 또는 특정 영역의 문자 인식(ex. 보험 청구서의 개인정보 및 청구항목 등)
- 반도체 검사 : 반도체 웨이퍼의 불량 검출, 반도체 적층 구조 얼라인 일치

■ 음성 / 자연어

- 음성 챗봇 : 고객 ARS 전화 수신, 고객 마케팅 전화 발신
- AI 스피커 : 고객 질문 이해, 언어 번역

■ 융합

- 콘텐츠 : 가상 캐릭터 광고
- 라이브 방송 : 가상 아나운서의 뉴스/라이브쇼핑 진행 등
- 메타버스 : 메타버스 공간 내의 가상 캐릭터/음성/영상 등



이름 로지[ROZY]
본명 오로지[Rozy Oh]/(one & only)이라는 뜻의 순수 한글 이름
나이 영원한 22세
생일 8월 19일
신체 171cm / 52kg / 250mm
혈액형 O형
MBTI ENFP 재가발달한 활동가
관심사 세미여행 / 요가 / 러닝 / 패션 / 예뻐라이프

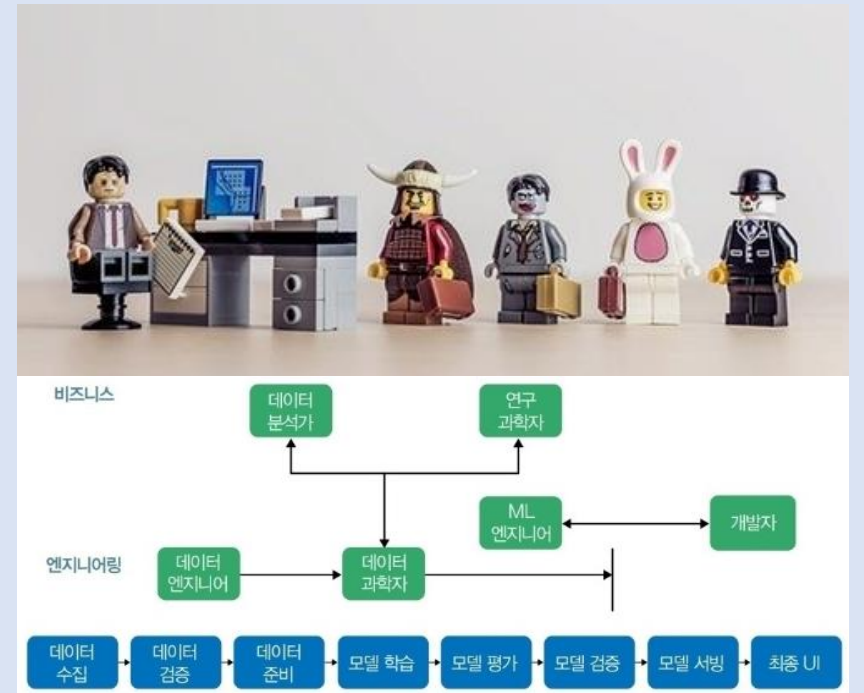
가상세계에서 태어나 현실세계와 소통하는
한국 최초의 바추얼인플루언서

패션과 라이프스타일의 트렌디함을 놓치지 않는 동시에
자연을 경이하고 탐험을 즐기며 환경을 생각하는 건강한 20대.

동양적인 외모와 서구적인 신체(자신이 좋아하는 것을 망설임 없이
추구하는 등 Gen Z 세대의 특징이 고스란히 녹아 있다)

3.1 딥러닝 연관 업무(개요)

- **데이터 과학자(data scientist)**
데이터셋 수집, 해석, 처리를 수행하는 직군으로 데이터에 대한 통계적, 탐색적 분석을 수행. 데이터 수집, 특징 가공, 모델 구축 등의 작업. 보통 파이썬 또는 R을 이용하여 조직의 머신러닝 모델을 가장 먼저 구축.
- **데이터 엔지니어 (data engineer)**
조직의 데이터를 위한 인프라와 워크플로를 관리하며 회사가 데이터를 수집하고, 파이프라인을 구축하고, 저장하고 전송하는 방법을 관리하는 데 도움. 데이터 엔지니어는 데이터를 중심으로 인프라와 파이프라인을 구현.
- **머신러닝 엔지니어 (machine learning engineer)**
데이터 과학자가 개발한 모델을 가져와서 해당 모델의 학습, 배포와 관련된 인프라와 운영을 관리. ML 엔지니어는 모델을 업데이트하고, 버전을 관리하고, 최종 사용자에게 예측 서비스를 처리하는 프로덕션 시스템을 구축.
- **연구 과학자 (research scientist)**
ML 분야를 발전시키기 위해 새로운 알고리즘을 찾고 개발하는 역할을 하며 연구 분야에는 모델 아키텍처, 자연어 처리, 컴퓨터 비전, 하이퍼파라미터 튜닝, 모델 해석과 같은 머신러닝 내의 다양한 하위 분야 포함. 다른 직군과는 달리 프로덕션 ML 시스템을 구축하는 대신, 새로운 접근 방식을 프로토타이핑하고 평가하는 데 대부분의 시간을 보냄.
- **데이터 분석가 (data analyst)**
데이터를 분석 및 인사이트를 도출하고 조직 내의 다른 팀에 공유. SQL, 스프레드시트에서 작업하고 비즈니스 인텔리전스 도구를 사용 데이터 시각화 결과 공유. 제품 팀과 긴밀히 협력하며, 보통 기존 데이터의 추세를 식별하고 통찰한 결과를 만들어내거나 향후 예측을 만들고 확장하는 업무.
- **ML 개발자 (machine learning developer)**
최종 사용자가 ML 모델에 접근할 수 있는 프로덕션 시스템 구축. 웹 또는 앱을 통해 사용자 친화적인 형식으로 모델에 쿼리를 날리고 예측을 반환받기 위한 API를 설계. API는 클라우드에서 호스팅되는 모델이거나 온디바이스로 내장된 모델. ML 엔지니어가 구현한 모델 서빙 인프라를 활용하여 사용자에게 결과 표시를 위한 애플리케이션과 사용자 인터페이스를 구축.



3.2 딥러닝 연관 업무(현업)

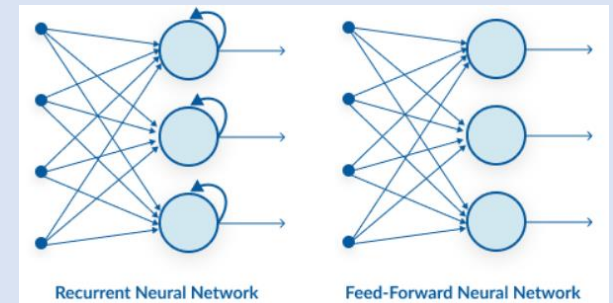
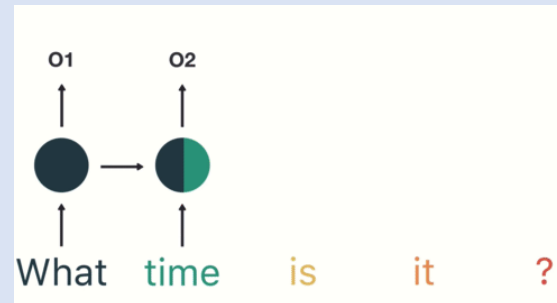
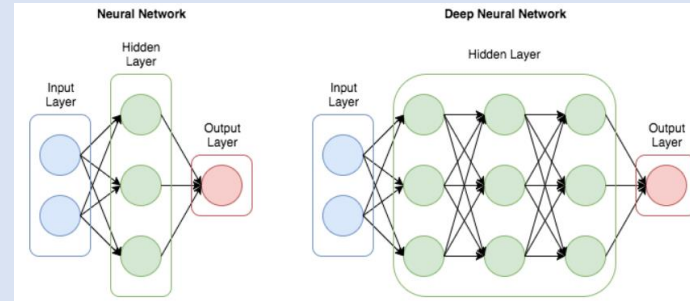
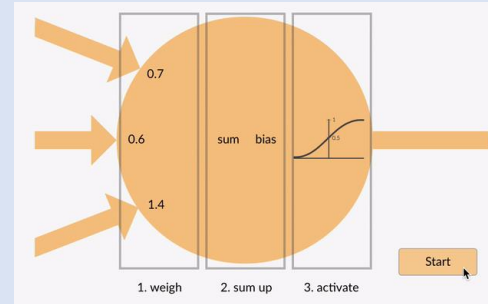
현업의 딥러닝 연관 업무

- 데이터 개발 및 관리
 - 업무 요약 : 데이터 수집 및 정제, 데이터 레이블링 및 관리를 주도하는 역할
 - 주요 업무 : 분야별 딥러닝 데이터에 대한 이해 필요, 레이블링 툴 사용 및 커스터 마이징, 외부 레이블링 데이터 검수, 딥러닝을 적용한 데이터 레이블링 자동화 또는 반자동화
 - 어려운 점 : 반복적 업무가 많음, 딥러닝에 대한 기본적인 이해가 필요함
- 모델 설계 및 학습
 - 업무 요약 : 기존 딥러닝 네트워크 모델을 활용하거나 새로운 모델을 설계하고, 필요한 학습을 진행하며, 결과를 도출하는 역할
 - 주요 업무 : 데이터 분석을 통한 네트워크 선정 또는 개발, 전처리를 통한 데이터 최적화, 학습 최적화, 추론 속도 최적화
 - 어려운 점 : 딥러닝을 위한 전반적인 사항을 이해하고 매니징해야 하며, 실제 딥러닝 관련 업무 외에도 최적화를 위한 다양한 업무 진행이 가능해야 함
- 학습 모델 배포 및 학습/배포 자동화(Devops)
 - 업무 요약 : 학습된 모델을 API 형태로 랩핑하여 사용자가 사용하기 쉬운 형태로 개발함
 - 주요 업무 : API 서버 개발, Request/Result 규약 정의, 서버 배포 및 이슈 대응, 학습 및 배포에 대한 자동화 인프라 개발
 - 어려운 점 : 딥러닝 프레임워크가 동작할 수 있는 패키지 개발이 필요하며, 속도나 이슈에 민감한 경우 시스템 의존성과 에러 처리에 주의 해야 함

4.1.1 딥러닝의 다양한 네트워크

딥러닝 네트워크 종류

- ANN(Artificial Neural Network) - 인공신경망
 - 모든 비선형 함수를 학습
 - 모든 입력을 출력에 매핑하는 가중치를 학습할 수 있는 능력
 - 활성화 함수는 네트워크에 비선형 속성 도입으로 입력과 출력 사이 복잡한 학습을 하는데 도움
 - 학습과정에서 파라미터 최적값 찾기 어려움
 - Overfitting에 따른 문제
- DNN(Deep Neural Network) - 심층신경망
 - ANN 문제 해결위해 은닉층 확대
 - 2개 이상의 은닉층으로 학습(보통 Deep Learning은 3개 이상)
 - DNN을 응용하여 CNN, RNN, LSTM, GRU 발전
- RNN(Recurrent Neural Network) - 순환신경망
 - RNN은 입력 데이터에 있는 순차 정보 캡처
 - O1, O2, O3, O4는 현재 단어만 아니라 이전 단어에도 의존(과거 학습을 Weight를 통해 현재 학습에 반영)
 - 여러 단계에서 매개 변수 공유 → 훈련 매개 변수 감소 및 계산 비용 감소
 - 반복적이고 순차적인 데이터에 효과

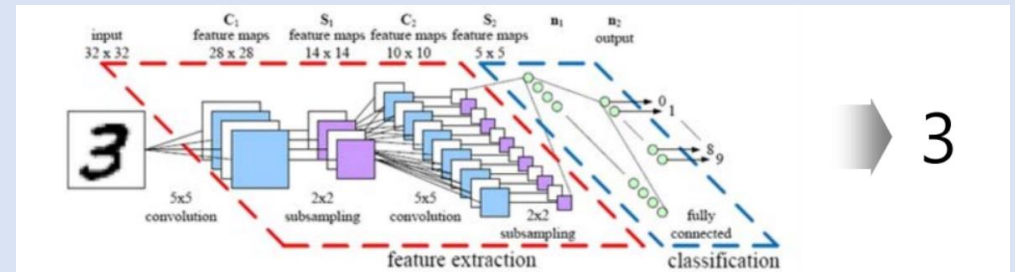
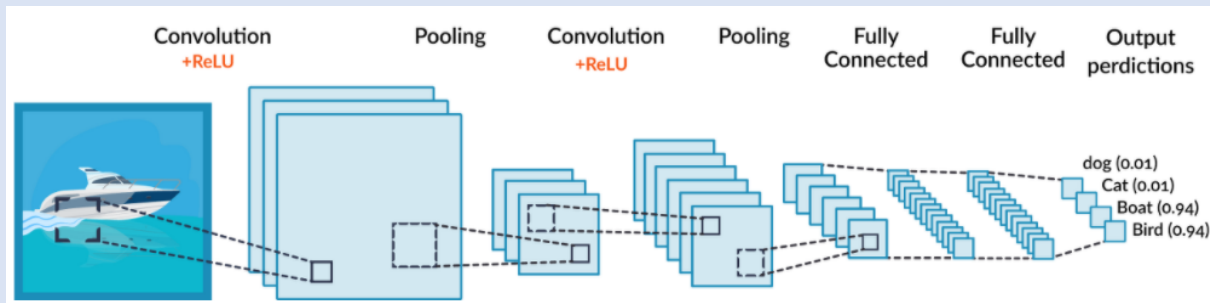
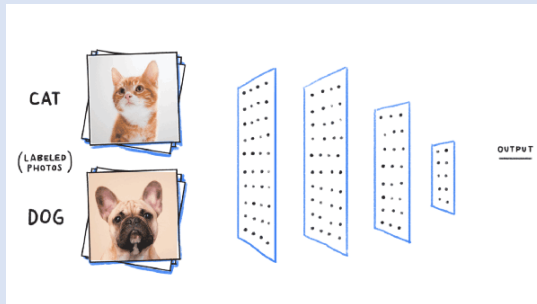


4.1.2 딥러닝의 다양한 네트워크

딥러닝 네트워크 종류

▪ CNN(Convolution Neural Network) - 합성곱신경망

- 정보추출, 문장분류, 얼굴인식 등 널리 사용, 특히 이미지 및 비디오 처리에 활용
- 핵심 요소는 커널(컨볼루션 연산을 사용하여 입력에서 관련 기능 추출)이라는 필터
- 암시적으로 필터를 자동으로 학습(입력데이터에서 올바른 관련 기능 추출에 도움)
- 입력 데이터의 특징을 추출하여 특징들의 패턴 파악하는 구조
- Convolution과정과 Pooling 과정으로 진행



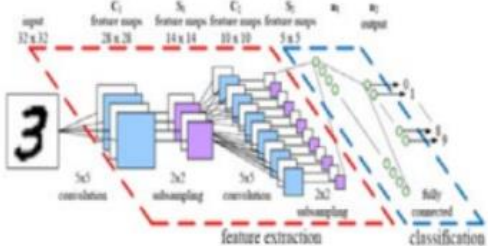
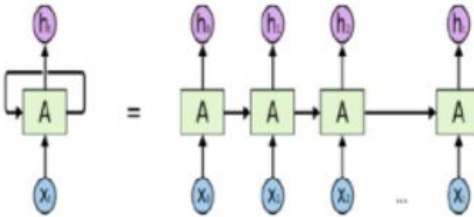
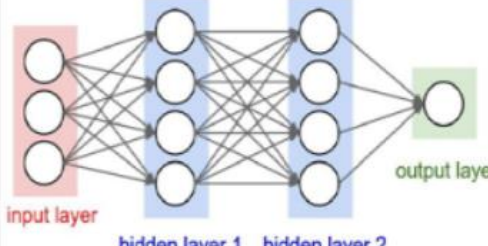
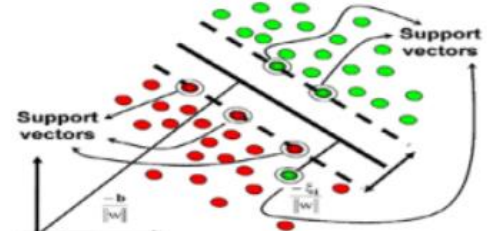
특징 추출 (Feature Extraction)

- ✓ Convolution layer
 - Pooling (Subsampling)
 - Activation function

분류 (Classification)

- ✓ Fully connected layer
 - Softmax
 - Dropout

4.2 딥러닝의 다양한 네트워크(비교)

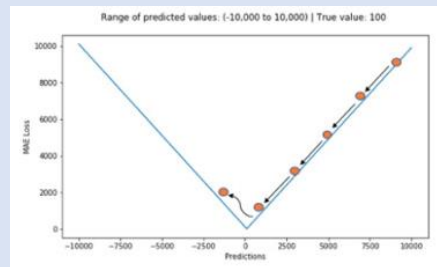
구분	CNN (Convolutional Neural Networks)	RNN (with Attention) (Recurrent Neural Network)	MLP (Multi-Layer Perceptron)	SVM (Support Vector Machine)
설명	<p>- 합성곱 계층을 통해 데이터의 특징 추출 후, 그 특징들을 기반으로 분류하는 딥러닝 분석모델</p> 	<p>- 데이터 순서정보를 반영하는 재귀 구조의 딥러닝 분석모델</p> <p>- Attention: 특정 데이터에 주목해 모델의 성능을 향상시키는 기법</p> 	<p>- 퍼셉트론(인간의 신경계 모방)을 여러 계층으로 조합한 분석모델</p> <p>- 딥러닝 알고리즘의 출발점</p> 	<p>- 주요 데이터(Support vectors) 선정을 통해 데이터를 구분하는 분석모델</p> 
장점	<ul style="list-style-type: none"> 타 알고리즘 대비 빠른 학습속도 데이터 분할 분석을 통한 오버피팅(과적합) 가능성 감소 텍스트 분류 시, 다수의 단어 조합에 대한 패턴 고려 가능 	<ul style="list-style-type: none"> 시계열 정보(텍스트 등) 반영 가능 입-출력 개수에 따른 다양한 모델 구성 가능 (1-1, 1-N, N-1, N-N) Attention: 학습 결과에 대한 (주요 활용 데이터) 시각화 가능 	<ul style="list-style-type: none"> 타 분석모델 대비 간단한 구조 타 분석모델들과의 조합을 통해, 다양한 분야에서 활용 가능 	<ul style="list-style-type: none"> 타 분석모델 대비 빠른 학습속도 데이터 특징(=컬럼)의 개수에 큰 영향 없이 균일한 성능 발휘
단점	<ul style="list-style-type: none"> 하이퍼 파라미터 설정에 따라, 과도한 데이터 손실에 따른 언더피팅 발생 가능 	<ul style="list-style-type: none"> 입력 데이터 전처리 필수 (텍스트 임베딩 등) Attention: 컴퓨팅 자원 추가소모 	<ul style="list-style-type: none"> 모델의 복잡도(계층 수 등) 증가 시, 오버피팅 발생 가능성 급증 	<ul style="list-style-type: none"> 입력 데이터 전처리 필수 (데이터 스케일 정규화)
주요 적용분야	<ul style="list-style-type: none"> 텍스트 분류 영상인식(탁월한 성능 발휘) 	<ul style="list-style-type: none"> 사진 설명, 텍스트 분류, 번역 등 	<ul style="list-style-type: none"> 범주형 데이터 분류 / 예측 	<ul style="list-style-type: none"> 데이터 분류 전반

5.1.1 딥러닝 손실 함수

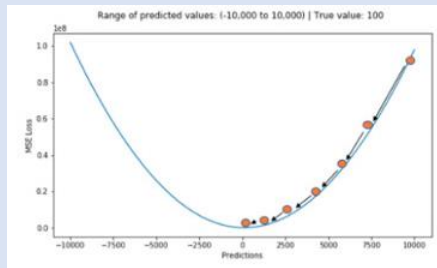
딥러닝의 손실함수 이해(1)

- 평균 절대 오차 (Mean Absolute Error, MAE)
평균 절대 오차는 예측 값과 정답 값의 차이에 절댓값을 취하여, 그 값들을 전부 더하고, 개수로 나누어 평균을 낸 값입니다. 예측 결과와 정답 결과가 떨어진 정도의 절댓값을 평균 낸 것이기에, 전체 데이터의 학습된 정도를 쉽게 파악할 수 있습니다. 하지만, 절댓값을 취하기 때문에 해당 예측이 어떤 식으로 오차가 발생했는지, 음수인지 양수인지 판단할 수 없다는 단점이 있습니다. 또한 아래 그림처럼 최적 값에 가까워지더라도 이동거리가 일정하기 때문에 최적 값에 수렴하기 어렵습니다.
- 평균 제곱 오차 (Mean Squared Error, MSE)
평균 제곱 오차는 가장 많이 쓰이는 손실 함수 중 하나이며, 예측 값과 실제 값 사이의 평균을 제곱하여 평균을 낸 값입니다. 차이가 커질수록 제곱 연산으로 인해 값이 뚜렷해지며 제곱으로 인해 오차가 양수이든 음수이든 누적 값을 증가시킵니다. MSE는 실제 정답에 대한 정답률의 오차뿐만 아니라 다른 오답들에 대한 정답률 오차 또한 포함하여 계산한다는 것이 특징입니다. MSE는 MAE와 달리 최적 값에 가까워질 경우 이동거리가 다르게 변화하기 때문에 최적 값에 수렴하기 용이합니다. MSE의 단점으로는 값을 제곱하기 때문에, "1 미만의 값은 더 작아지고, 그 이상의 값은 더 커진다", 즉 값의 왜곡이 있을 수 있습니다.
- 평균 제곱근 오차(Root Mean Square Error, RMSE)
평균 제곱근 오차는 MSE에 루트를 씌운 지표로 장단점은 MSE와 유사한 형태로 이루어집니다. 하지만 제곱된 값에 루트를 씌우기 때문에 값을 제곱해서 생기는 왜곡이 줄어들며, 오차를 보다 직관적으로 보여줍니다. 그 이유는 루트를 씌워주기 때문에 오류 값을 실제 값과 유사한 단위로 변환하여 해석을 할 수 있기 때문입니다.

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



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



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

5.1.2 딥러닝 손실 함수

딥러닝의 손실함수 이해(2)

Cross-entropy는 실제 분포 q 에 대하여 알지 못하는 상태에서, 모델링을 통하여 구한 분포인 p 를 통하여 q 를 예측하는 것입니다. 이때, q 와 p 가 모두 식에 들어가기 때문에, cross-entropy라는 이름이 생겼습니다.

Cross-entropy에서 실제 값과 예측 값이 맞는 경우에는 0으로 수렴하고, 값이 틀릴 경우에는 값이 커지기 때문에, 실제 값과 예측 값의 차이를 줄이기 위한 방식이라고 이해하시면 됩니다. 하지만 이 Cross-entropy는 label의 값이 one-hot encoding일 경우에만 사용 가능합니다.

One hot encoding: multinomial classification에서 사용하는 인코딩 방법으로 출력 값의 형태가 정답 label은 1이고 나머지 label 값은 모두 0 ex) [1,0,0], [0,1,0], [0,0,1]

- Binary cross-entropy

Binary cross-entropy는 이진 분류에 사용되는 방식입니다. 예를 들어, True 또는 False, 양성 또는 음성 등 2개의 class를 분류할 때 사용하는 방식으로 예측값이 0과 1 사이의 확률 값으로 나옵니다. 1에 가까우면 하나의 클래스(True or 양성)일 확률이 큰 것이고, 0에 가까우면 다른 하나의 클래스(False or 음성)일 확률이 큰 것입니다.

- Categorical cross-entropy

Categorical cross-entropy는 분류해야 할 클래스가 3개 이상인 경우, 즉 멀티클래스 분류에 사용됩니다. 라벨이 [0,0,1,0,0], [1,0,0,0,0], [0,0,0,1,0]과 같이 one-hot 형태로 제공될 때 사용됩니다. 일반적으로 예측 값은 [0.02 0.94 0.02 0.01 0.01]와 같은 식으로 나오기 때문에 여러 class 중 가장 적절한 하나의 class를 분류하는 문제의 손실 함수로 사용되기에 적합합니다.

$$H_p(q) = - \sum_{c=1}^C q(y_c) \log(p(y_c))$$

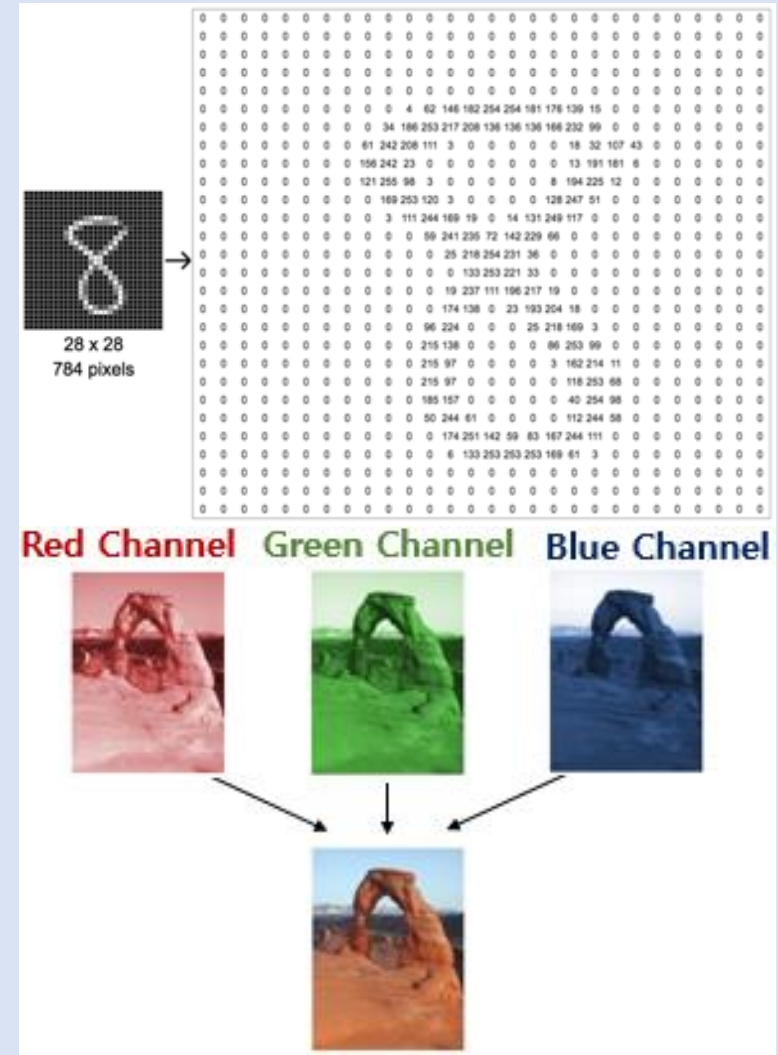
$$BCE = - \frac{1}{N} \sum_{i=0}^N y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)$$

$$CCE = - \frac{1}{N} \sum_{i=0}^N \sum_{j=0}^J y_j \cdot \log(\hat{y}_j) + (1 - y_j) \cdot \log(1 - \hat{y}_j)$$

6.1 영상 분야의 딥러닝을 위한 사전 지식

이미지의 이해(해상도/픽셀/채널)

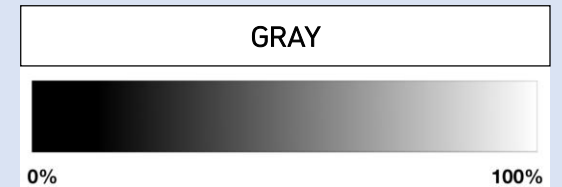
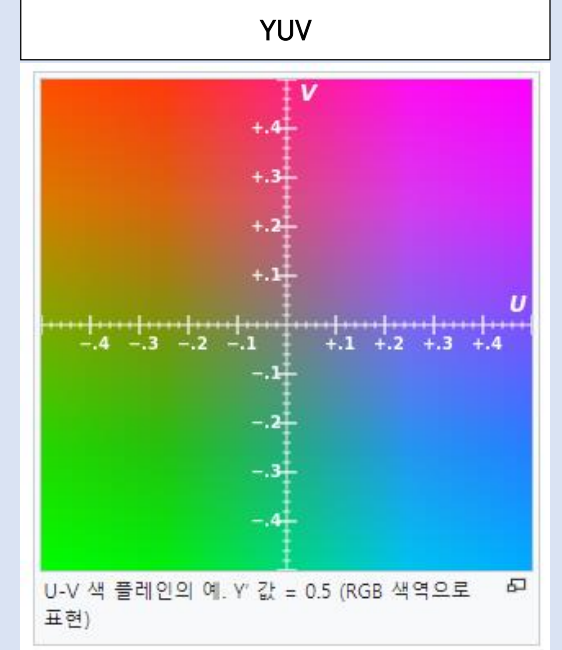
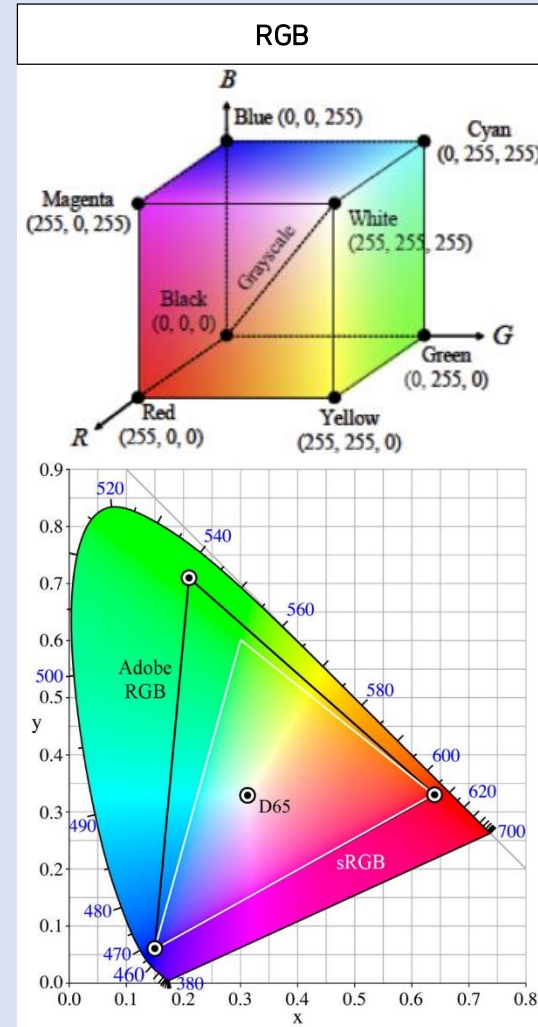
- 해상도
이미지의 가로 세로 크기
- 픽셀
이미지가 가로 세로 크기가 이루는 한 점의 단위
- 채널
이미지의 한 점의 단위에서 색상을 나타내기 위한 단위



6.2.1 영상 분야의 딥러닝을 위한 사전 지식

이미지의 이해(색공간1)

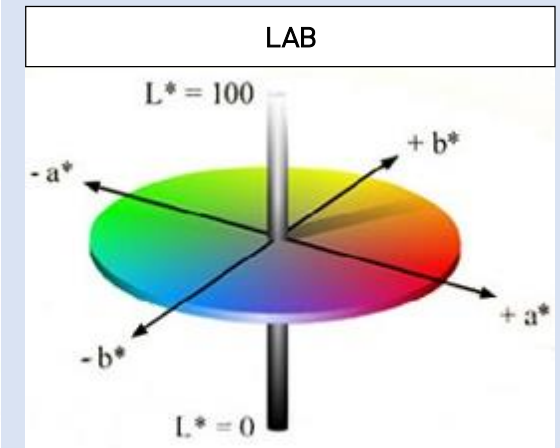
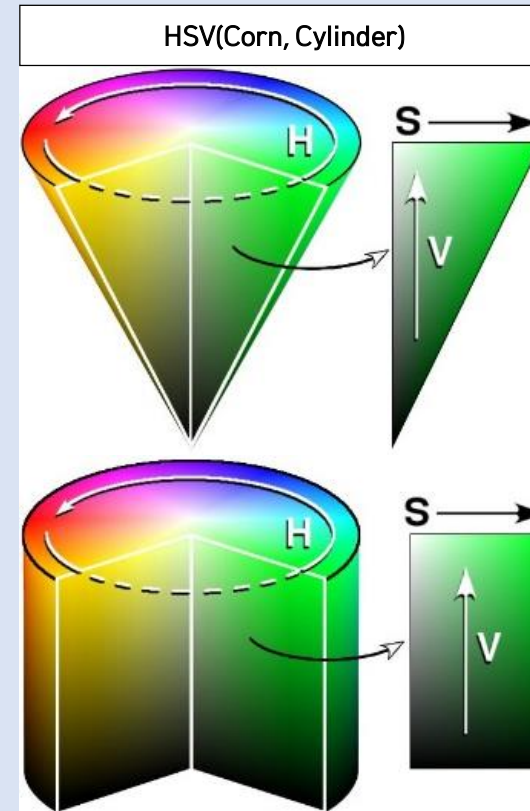
- **RGB(sRGB)**
RGB 색 공간은 색을 혼합하면 명도가 올라가는 가산 혼합 방식으로 색을 표현한다. RGB 가산혼합의 삼원색은 빨강(Red), 녹색(Green)[1], 파랑(Blue)을 뜻한다. RGBA은 RGB와 동일하며, 알파(Alpha)라는 투과도를 덧붙인 것이다. RGB 색 공간은 삼원색에 해당하는 세 가지 채널의 밝기를 기준으로 색을 지정한다. RGB 색 공간은 웹 색상 표현의 기본 원리이다.
- **YUV(YCbCr)**
YUV(YIQ/YCbCr/YPbPr)는 색을 구성하는 방법 중 하나로 휘도(Y)와 청색 색차(U), 적색 색차(V) 정보로 색을 구성한다. 여기서 Y 신호만 받는다면 흑백이 된다. 여러 번 복제된 VHS 테이프나 방송 상태가 좋지 못한 채널에서 흑백으로 보이는 것도 이것 때문이다. Lab과 마찬가지로 인간이 색을 인식하는 방식으로 구성되었다.
- **GARY**
YUV 색공간에서 Y Channel 을 GRAY로 정의 하며, 색의 밝기만을 표시한다.(0: 검정색 ~ 255:흰색)



6.2.2 영상 분야의 딥러닝을 위한 사전 지식

이미지의 이해(색공간2)

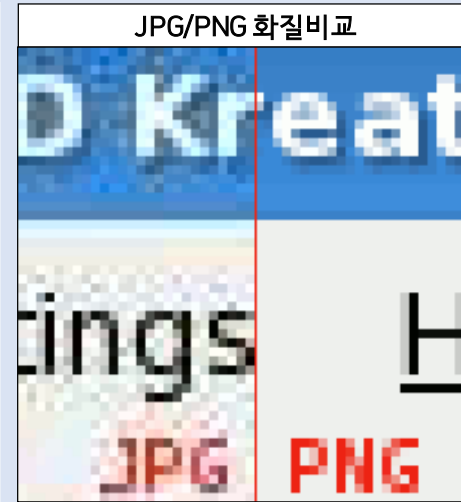
- **HSV**
HSV 색 공간은 색상(Hue), 채도(Saturation), 명도(value)를 기준으로 색을 구성하는 방식이다. 감산 혼합이나 가산 혼합보다 색상의 지정이 직관적이기 때문에 시각 예술에서 자주 쓰인다.
- **Lab(CIELab)**
CIE $L^*a^*b^*$ 색 공간에서 L^* 값은 밝기를 나타낸다. $L^* = 0$ 이면 검은색이며, $L^* = 100$ 이면 흰색을 나타낸다. a^* 은 빨강과 초록 중 어느쪽으로 치우쳤는지를 나타낸다. a^* 이 음수이면 초록에 치우친 색깔이며, 양수이면 빨강/보라 쪽으로 치우친 색깔이다. b^* 은 노랑과 파랑을 나타낸다. b^* 이 음수이면 파랑이고 b^* 이 양수이면 노랑이다.
또한, 인간의 색 지각이 비선형이라는 연구 결과에 따라, $L^*a^*b^*$ 색 공간은 실제 빛의 파장과 비선형적 관계를 갖는다. 또한 $L^*a^*b^*$ 공간에서 서로 다른 두 색의 거리는 인간이 느끼는 색깔의 차이와 비례하도록 설계되었다.



6.3 영상 분야의 딥러닝을 위한 사전 지식

디지털 이미지의 이해(이미지 압축 포맷)

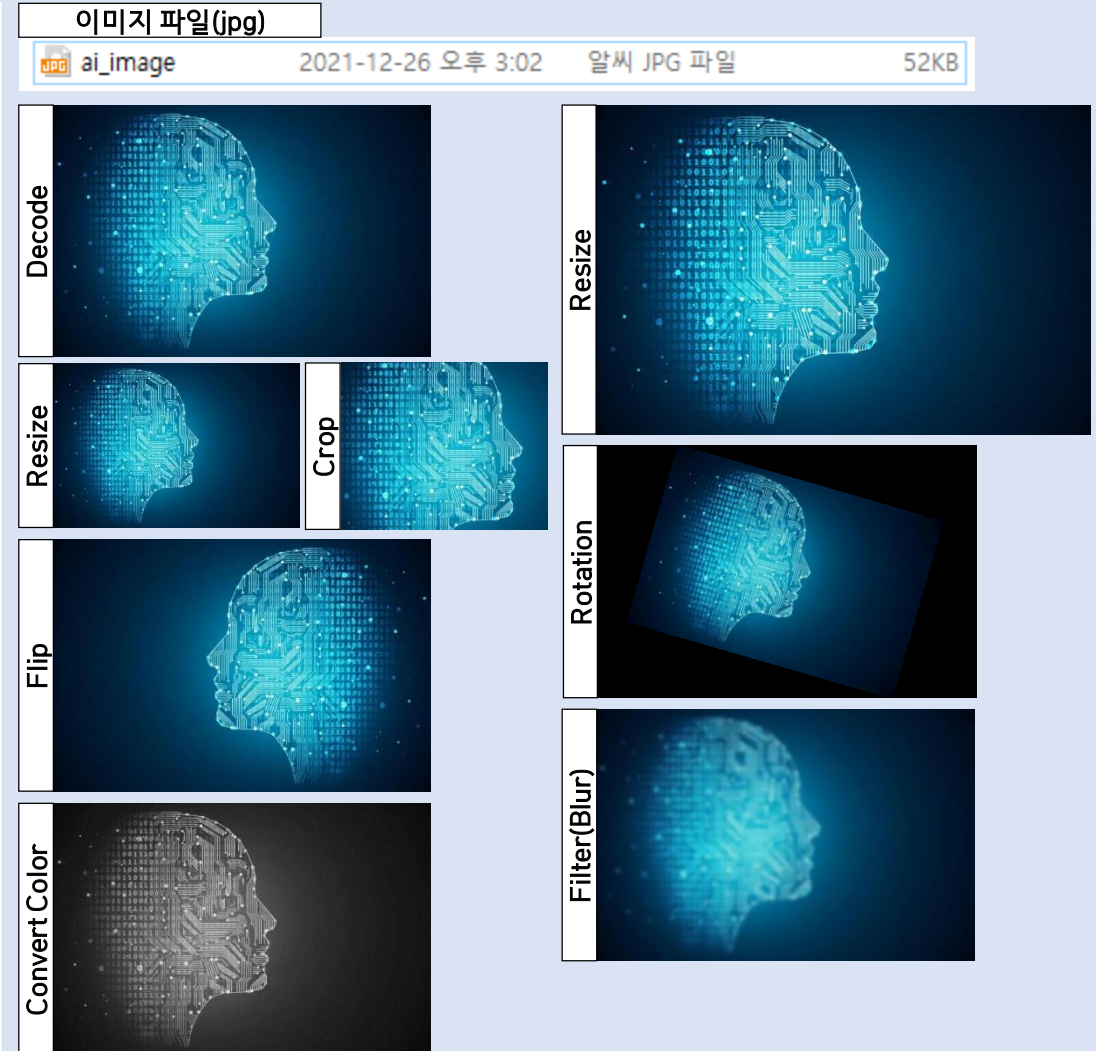
- **BMP**
일반적으로 이미지를 압축하지 않고, Header 정보와 RGB Buffer를 저장한 파일로 이미지 해상도에 따라 파일의 크기가 커진다.
- **JPG**
이미지를 DCT와 Huffman 알고리즘을 이용하여 압축하여 저장한 형태로 일반적으로 압축 손실이 발생하여 디코딩시 원본과 차이가 발생하지만, 압축된 파일 크기가 작은 특징을 갖는다.
- **PNG**
이미지를 무손실로 압축하는 방식으로 투명 채널을 지원하며, JPEG 보다는 대체로 파일 크기가 크지만 BMP에 비해서는 훨씬 작은 파일 크기를 갖는다.
- **GIF**
이미지를 256컬러로 양자화 한 다음에 압축하는 방식으로 파일크기가 작아 보이지만 색상의 손실이 많이 발생하고, 대체로 PNG 보다 큰 파일 크기를 갖는다. 장점으로는 움직이는 이미지를 지원한다.



6.4 영상 분야의 딥러닝을 위한 사전 지식

디지털 이미지의 이해(이미지 다루기 기본)

- **Decode**
압축된 이미지를 복원하여, 픽셀과 채널 형태를 구성하여 메모리 Buffer에 담는 것
- **Resize**
이미지의 해상도를 키우거나 줄이는 것
- **Crop**
이미지의 일부 영역을 잘라 내는 것
- **Rotation**
이미지를 회전하는 것
- **Flip**
이미지를 좌/우 또는 상/하 반전 시키는 것
- **Color Convert**
RGB 이미지를 Gray이미지 또는 YUV 이미지 등으로 색공간을 변환하는 것
- **Filter**
이미지에 특정한 필터를 통해서 블러 처리를 하거나 선명하게 하는 등의 처리



6.5 영상 분야의 딥러닝을 위한 사전 지식

디지털 이미지의 이해(이미지 다루기 심화)

- **Affine Transform**

Affine 변환은 직선, 길이(거리)의 비, 평행성(parallelism)을 보존하는 변환이며, 회전, 평행이동, 스케일 뿐만 아니라 shearing, 반전(reflection)까지를 포함한 변환이다.

- **Perspective Transform**

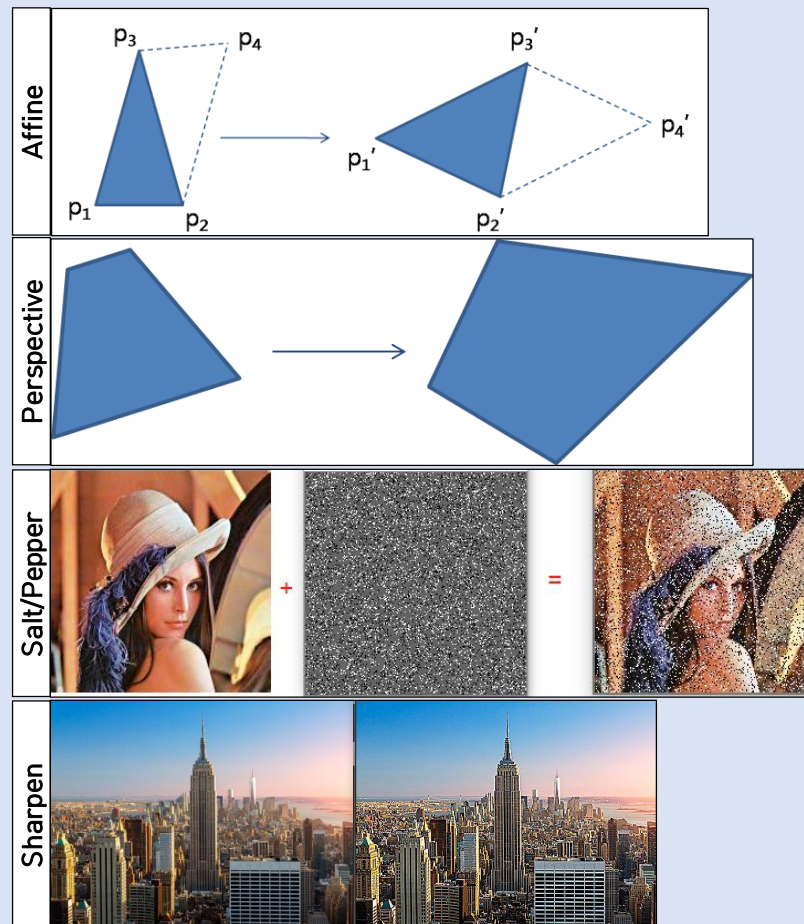
Planer surface 물체의 경우에는 3D 공간에서의 2D 이미지로의 임의의 원근투영변환(perspective projective transformation)을 두 이미지 사이의 homography로 모델링할 수 있습니다. 즉, 어떤 planar surface가 서로 다른 카메라 위치에 대해 이미지 A와 이미지 B로 투영되었다면 이미지 A와 이미지 B의 관계를 homography로 표현할 수 있다는 것입니다. 그래서 homography는 평면물체의 2D 이미지 변환관계를 설명할 수 있는 가장 일반적인 모델이며, 이를 perspective(projective) transformation과 같이 부릅니다.

- **Salt/Pepper Noise**

이미지 상의 랜덤한 픽셀의 원래 값을 검정색(0)과 흰색(255) 값으로 변경하여 처리하는 것

- **Sharpen**

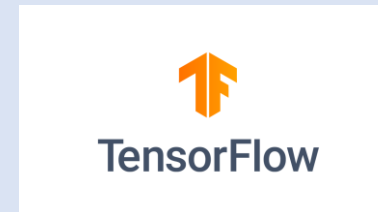
이미지의 색상을 주변 색상과 대비를 크게 만들어서 선명하게 보이도록 처리하는 것



7.1 딥러닝 기술 개발을 위한 준비

딥러닝 기술 개발을 위한 준비 환경

- 개발 머신 준비
 - Linux, Windows OS 탑재된 GPU 머신, 또는 Cloud 가상 자원
- 개발환경 구축
 - Docker, Anaconda, Python, CUDA 설치
- 필요 패키지 설치
 - Numpy, OpenCV, Pytorch(또는 Tensorflow), JupyterLab, Tensorboard



7.2 딥러닝 기술 개발을 위한 준비

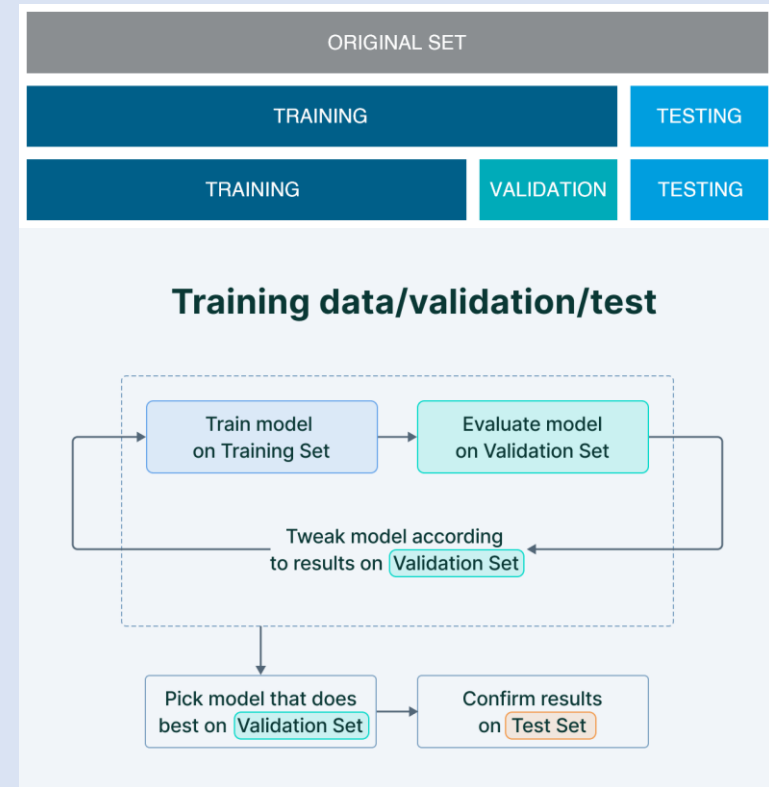
딥러닝 기술 개발을 위한 준비

▪ 데이터

- Training Set
학습을 위한 데이터 셋으로 실제 학습에 사용되는 데이터로 일반적인 데이터로 많은 양의 데이터를 구성해야 하며 데이터 레이블 필요
- Validation Set
학습 중인 모델의 학습률 평가를 위한 데이터 셋으로 실제 학습에는 포함되지 않으며, 학습 목적이 요구하는 일반적인 데이터에 대한 평가가 가능할 수 있도록 데이터를 구성해야 하며, 데이터 레이블 필요
- Testing Set
최종 학습 모델의 결과를 평가하기 위한 데이터

▪ 학습 코드

- Dataset/DataLoader : 데이터를 불러오고, 모델에 적재하기 위한 코드
- Model : 학습의 구조 정의와 가중치를 업데이트 하기 위한 코드
- Loss Function : 학습에서 정답과 예측치의 차이 계산을 통해 가중치를 계산하기 위한 코드
- Optimizer : Loss로 부터 해당 값을 최소화 하기 위한 최적화 방식이 포함된 코드
- Train/Evaluate : 학습 과정과 검증 과정을 관리하기 위한 코드
- Test : 최종 학습 후 모델을 로드하여 실제 사용하기 위한 코드




8.1 딥러닝 실습 환경 구축

실습 환경 구축

- CUDA 설치
 - CUDA Compatability : <https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html>
 - CUDA : <https://developer.nvidia.com/cuda-toolkit-archive>
 - CUDNN : <https://developer.nvidia.com/cudnn>
- 가상환경 설치
 - Anaconda 설치 : <https://www.anaconda.com/products/individual>
- 딥러닝 프레임워크
 - Pytorch : <https://pytorch.org/get-started/locally/>
 - Tensorflow : <https://www.tensorflow.org/install?hl=ko>

Anaconda Individual Edition

Download 

For Windows

Python 3.9 • 64-Bit Graphical Installer • 510 MB

START LOCALLY

Select your preferences and run the install command. Stable represents the most currently tested and supported version of PyTorch. This should be suitable for many users. Preview is available if you want the latest, not fully tested and supported, 1.11 builds that are generated nightly. Please ensure that you have met the prerequisites below (e.g., numpy), depending on your package manager. Anaconda is our recommended package manager since it installs all dependencies. You can also install previous versions of PyTorch. Note that LibTorch is only available for C++.

Additional support or warranty for some PyTorch Stable and LTS binaries are available through the PyTorch Enterprise Support Program.

PyTorch Build	Stable (1.10.1)	Preview (Nightly)	LTS (1.8.2)
Your OS	Linux	Mac	Windows
Package	Conda	Pip	LibTorch
Language	Python	C++ / Java	
Compute Platform	CUDA 10.2	CUDA 11.3	ROCm-6.0 (beta)
Run this Command:	pip3 install torch==1.10.1+cu113 torchvision==0.11.2+cu113 torchaudio==0.10.1+cu113 -f https://download.pytorch.org/whl/cu113/torch_stable.html		

Table 2. CUDA Toolkit and Minimum Required Driver Version for CUDA Minor Version Compatibility

CUDA Toolkit	Minimum Required Driver Version for CUDA Minor Version Compatibility*	
	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 11.5.x	>=450.80.02	>=452.39
CUDA 11.4.x	>=450.80.02	>=452.39
CUDA 11.3.x	>=450.80.02	>=452.39
CUDA 11.2.x	>=450.80.02	>=452.39
CUDA 11.1 (11.1.0)	>=450.80.02	>=452.39
CUDA 11.0 (11.0.3)	>=450.36.06**	>=451.22**

Table 3. CUDA Toolkit and Corresponding Driver Versions

CUDA Toolkit	Toolkit Driver Version	
	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 11.5 Update 1	>=495.29.05	>=496.13
CUDA 11.5 GA	>=495.29.05	>=496.04
CUDA 11.4 Update 3	>=470.82.01	>=472.50
CUDA 11.4 Update 2	>=470.57.02	>=471.41
CUDA 11.4 Update 1	>=470.57.02	>=471.41
CUDA 11.4.0 GA	>=470.42.01	>=471.11
CUDA 11.3.1 Update 1	>=465.19.01	>=465.89
CUDA 11.3.0 GA	>=465.19.01	>=465.89
CUDA 11.2.2 Update 2	>=460.32.03	>=461.33
CUDA 11.2.1 Update 1	>=460.32.03	>=461.09
CUDA 11.2.0 GA	>=460.27.03	>=460.82

Select Target Platform

Click on the green buttons that describe your target platform. Only supported platforms will be shown. By downloading and using the software, you agree to fully comply with the terms and conditions of the CUDA EULA.

Operating System

Linux Windows

Architecture

x86_64

Version

10 Server 2016 Server 2019

Installer Type

exe (local) exe (network)

Download installer for Windows 10 x86_64

The base installer is available for download below.

> Base Installer Download (37.8 MB)

cuDNN Archive

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

Download cuDNN v8.3.0 (November 3rd, 2021), for CUDA 11.5

Download cuDNN v8.3.0 (November 3rd, 2021), for CUDA 10.2

Download cuDNN v8.2.4 (September 2nd, 2021), for CUDA 11.4

Download cuDNN v8.2.4 (September 2nd, 2021), for CUDA 10.2

Download cuDNN v8.2.2 (July 6th, 2021), for CUDA 11.4

Download cuDNN v8.2.2 (July 6th, 2021), for CUDA 10.2

Download cuDNN v8.2.1 (June 7th, 2021), for CUDA 11.x

Library for Windows and Linux, Ubuntu(x86_64, armv8b, PPC architecture)

cuDNN Library for Linux (aarch64b)

cuDNN Library for Linux (x86_64)

cuDNN Library for Linux (PPC)

cuDNN Library for Windows (x86)

8.2 딥러닝 실습 환경 구축

실습 환경 동작 확인

- Anaconda 확인
 - 설치
c:\> conda --version
 - 동작
c:\> conda env list
c:\> conda list
- CUDA 확인
 - 설치("C:\Program Files\NVIDIA Corporation\NVSMI\")
c:\> nvidia-smi
c:\> nvcc --version

```
C:\Users\nick>conda --version
conda 4.10.3

C:\Users\nick>conda env list
# conda environments:
#
base                  * C:\Users\nick\anaconda3

C:\Users\nick>conda list
# packages in environment at C:\Users\nick\anaconda3:
#
# Name                    Version            Build                Channel
#-----
# _ipyw_jlab_nb_ext_conf  0.1.0              py39haa95532_0      py39haa95532_0
# alabaster                0.7.12            pyhd3eb1b0_0        py39_0
# anaconda                 2021.11           py39haa95532_0      py39_0
# anaconda-client          1.9.0             py39haa95532_0      py39_0
# anaconda-navigator       2.1.1             py39_0
# anaconda-project         0.10.1            pyhd3eb1b0_0        py39_0
# anvio                    2.2.0             py39haa95532_2      py39haa95532_2
# appdirs                  1.4.4             pyhd3eb1b0_0        py39_0
# argh                     0.26.2            py39haa95532_0      py39haa95532_0
# argon2-cffi              20.1.0            py39h2bfff1b_1      py39h2bfff1b_1
# arrow                    0.13.1            py39haa95532_0      py39haa95532_0
```

```
C:\Users\nick>nvidia-smi
Sun Dec 26 12:01:30 2021

+-----+
| NVIDIA-SMI 462.80              Driver Version: 462.80          CUDA Version: 11.2     |
+-----+-----+
| GPU Name   TCC/WDDM  Bus-Id  Disp.A   Volatile Uncorr. ECC    |
| Fan  Temp  Perf  Pwr:Usage/Cap  Memory-Usage  GPU-Util  Compute M. |
| N/A    N/A    P8     8W /  N/A   161MiB /  8192MiB      0%      Default  |
+-----+-----+

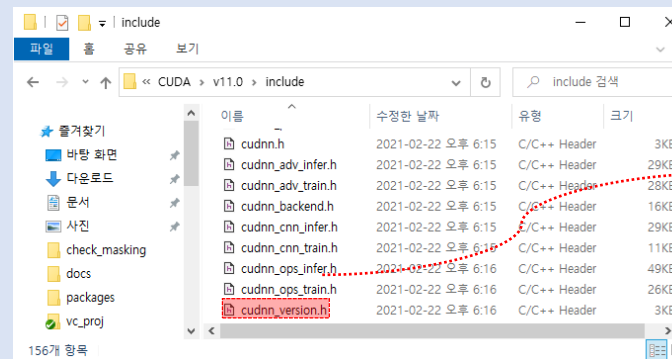
Processes:
GPU  GI  CI  PID  Type  Process name                        GPU Memory
ID   ID   ID                   Usage
--  --  --  --  --  --
0   N/A  N/A  1476  C+G   Insufficient Permissions            N/A

C:\Users\nick>

C:\Users\nick>nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2020 NVIDIA Corporation
Built on Thu_Jun_11_22:26:48_Pacific_Daylight_Time_2020
Cuda compilation tools, release 11.0, V11.0.194
Build cuda_11.0_bu.relgpu_drvr445TC445_37.28540450_0

C:\Users\nick>
```

용어	설명
Driver Version	GPU의 버전
CUDA Version	11.2가 설치되어있는 것이 아닌 11.2를 설치해야 한다는 뜻 !
GPU / Fan	GPU Number와 Fan의 사용 %를 나타냄
Name	GPU MODEL
Temp	GPU가 일정 온도가 지나면 성능이 저하되는 것을 알려줌
Perf	P0 ~ P12를 가지며, 숫자가 적을수록 높은 성능



```
cudnn_version.h - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
/*
 * @file: The master cudNN version file.
 */

#ifndef CUDNN_VERSION_H_
#define CUDNN_VERSION_H_

#define CUDNN_MAJOR 8
#define CUDNN_MINOR 1
#define CUDNN_PATCHLEVEL 1

#define CUDNN_VERSION (CUDNN_MAJOR * 1000 + CUDNN_MINOR * 100 + CUDNN_PATCHLEVEL)

#endif /* CUDNN_VERSION_H_ */
```

8.3 딥러닝 실습 환경 구축

가상환경 생성 및 추가 패키지 설치

■ Anaconda 가상환경 생성 및 가상환경 구동

- 가상환경 생성 및 확인
c:\> conda search python
c:\> conda create -n torch_env python=3.8
c:\> conda env list
- 가상환경 구동
c:\> conda activate torch_env

■ 추가 패키지 설치

- Numpy, Matplotlib
(torch_env) c:\> pip install numpy matplotlib
- Opencv
(torch_env) c:\> pip install opencv-python
- Jupyterlab, Tensorboard
(torch_env) c:\> pip install jupyterlab tensorboard
- Pytorch
(torch_env) c:\> conda install pytorch torchvision torchaudio
cudatoolkit=11.3 -c pytorch

■ 패키지 설치 확인

- PIP
(torch_env) c:\> pip list
- CONDA
(torch_env) c:\> conda list

```
done
# To activate this environment, use
#
#     $ conda activate torch_env
#
# To deactivate an active environment, use
#
#     $ conda deactivate
#

C:\Users\nick>conda env list
# conda environments:
#
base                  *  C:\Users\nick\anaconda3
torch_env             C:\Users\nick\anaconda3\envs\torch_env

C:\Users\nick>conda activate torch_env

(torch_env) C:\Users\nick>pip list
Package      Version
-----
certifi      2021.10.8
pip          21.2.2
setuptools   58.0.4
wheel        0.37.0
wincertstore 0.2

(torch_env) C:\Users\nick>conda list
# packages in environment at C:\Users\nick\anaconda3\envs\torch_env:
#
# Name                    Version           Build    Channel
# -----
anyio                     3.4.0            pypi_0  pypi
argon2-cffi              21.3.0            pypi_0  pypi
argon2-cffi-bindings    21.2.0            pypi_0  pypi
attrs                    21.2.0            pypi_0  pypi
Babel                    2.9.1            pypi_0  pypi
backcall                 0.2.0            pypi_0  pypi
bleach                   4.1.0            pypi_0  pypi
ca-certificates          2021.10.26       haa95532_2  pypi
certifi                  2021.10.8       py38haa95532_0  pypi
cffi                     1.15.0           pypi_0  pypi
charset-normalizer       2.0.9            pypi_0  pypi
colorama                 0.4.4            pypi_0  pypi
cudatoolkit              11.3.1           h59b6b97_2  pypi
cycl                      0.11.0           pypi_0  pypi
debugpy                  1.5.1            pypi_0  pypi
decorator                5.1.0            pypi_0  pypi
defusedxml               0.7.1            pypi_0  pypi
entrypoints              0.3              pypi_0  pypi
fonttools                4.28.5           pypi_0  pypi
freetype                 2.10.4           hd928e21_0  pypi
idna                     3.3              pypi_0  pypi
importlib-resources      5.4.0            pypi_0  pypi
intel-openmp             2021.4.0         haa95532_3556  pypi
ipykernel                6.6.0            pypi_0  pypi
ipython                  7.30.1           pypi_0  pypi
ipython-genutils         0.2.0            pypi_0  pypi
```

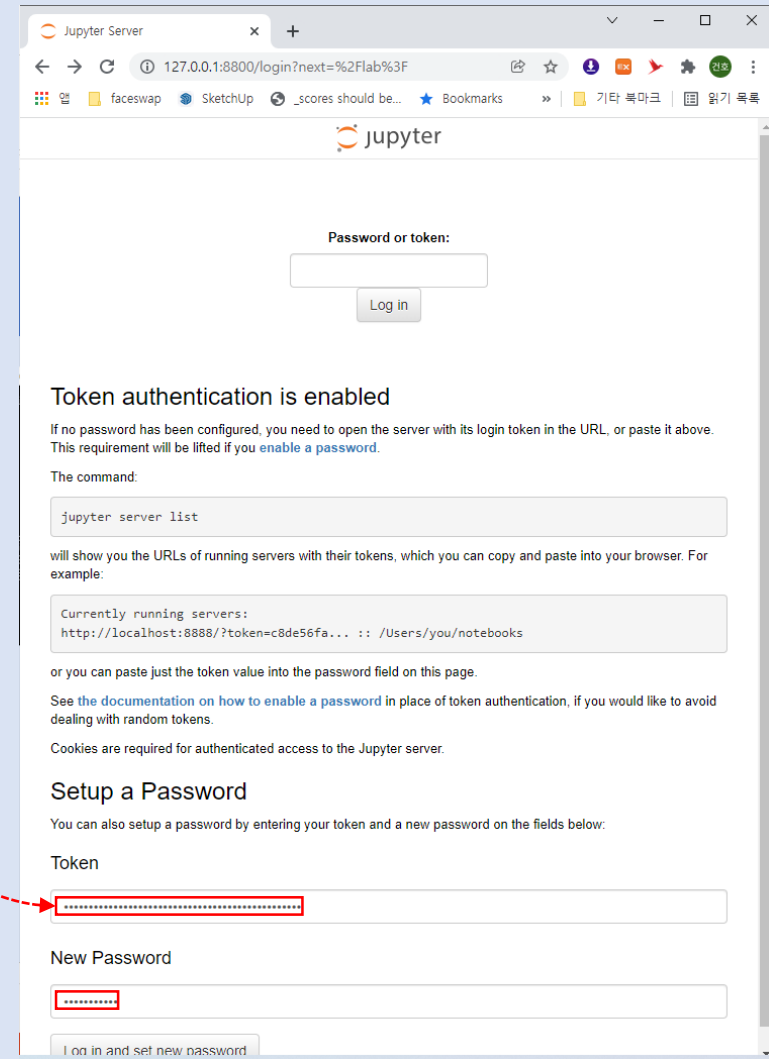

8.4 딥러닝 실습 환경 구축

Jupyterlab 개발 환경 구동

- Jupyter 서버 동작
 - Jupyter lab -ip=<ip_addr> --port=<port> --no-browser
(torch_env) c:\> jupyter lab -ip=0.0.0.0 -port=8800 --no-browser
- Jupyter 서버 접속
 - 인터넷 브라우저 : http://127.0.0.1:8800

```
명령 프롬프트 - conda install pytorch torchvision torchaudio cudatoolkit=11.3 -c pytorch - jupyter lab --ip=0.0.0.0 --port=8800 --no-browser
(torch_env) C:\Users\nick>
(torch_env) C:\Users\nick>jupyter lab --ip=0.0.0.0 --port=8800 --no-browser
[2021-12-26 13:49:22.437 ServerApp] jupyterlab | extension was successfully linked.
[2021-12-26 13:49:22.794 ServerApp] nbclassic | extension was successfully linked.
[2021-12-26 13:49:22.834 ServerApp] nbclassic | extension was successfully loaded.
[2021-12-26 13:49:22.835 LabApp] JupyterLab extension loaded from C:\Users\nick\Anaconda3\envs\torch_env\lib\site-packages\jupyterlab
[2021-12-26 13:49:22.835 LabApp] JupyterLab application directory is C:\Users\nick\Anaconda3\envs\torch_env\share\jupyterlab
[2021-12-26 13:49:22.839 ServerApp] jupyterlab | extension was successfully loaded.
[2021-12-26 13:49:22.840 ServerApp] Serving notebooks from local directory: C:\Users\nick
[2021-12-26 13:49:22.840 ServerApp] Jupyter Server 1.13.1 is running at:
[2021-12-26 13:49:22.841 ServerApp] http://DESKTOP-IG9C06F:8800/lab?token=3226ee191fdb59c0ab28e3bb3d4317b548756436c6b7d93b
[2021-12-26 13:49:22.841 ServerApp] or http://127.0.0.1:8800/lab?token=3226ee191fdb59c0ab28e3bb3d4317b548756436c6b7d93b
[2021-12-26 13:49:22.841 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[2021-12-26 13:49:22.853 ServerApp]

To access the server, open this file in a browser:
file:///C:/Users/nick/AppData/Roaming/jupyter/runtime/jpserver-18132-open.html
Or copy and paste one of these URLs:
http://DESKTOP-IG9C06F:8800/lab?token=3226ee191fdb59c0ab28e3bb3d4317b548756436c6b7d93b
or http://127.0.0.1:8800/lab?token=3226ee191fdb59c0ab28e3bb3d4317b548756436c6b7d93b
[2021-12-26 13:49:40.983 ServerApp] 302 GET / (127.0.0.1) 1.00ms
[2021-12-26 13:49:40.989 LabApp] 302 GET /lab? (127.0.0.1) 0.99ms
```



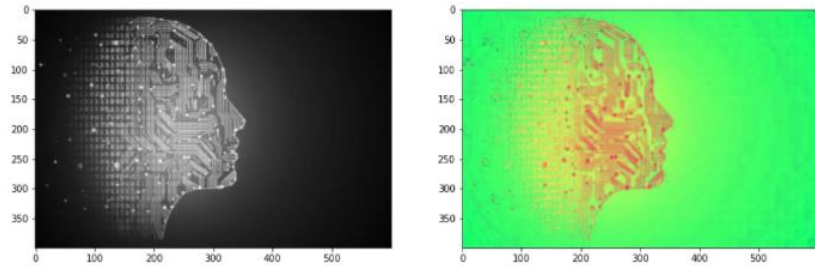
9.1 딥러닝 실습을 위한 사전 연습

영상처리

- **Package import**
import cv2
- **Image Read**
img = cv2.imread("img.jpg") //bgr order image
- **Image Resize**
h, w, ch = img.shape
new_size=(int(w*0.5), int(h*0.5))
scaledimg = cv2.resize(img, disze=new_size, fx=0, fy=0, interpolation=cv2.INTER_CUBIC)
scaledimg = cv2.resize(img, disze=(0, 0), fx=0.5, fy=0.5, interpolation=cv2.INTER_CUBIC)
- **Image Color Transform**
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

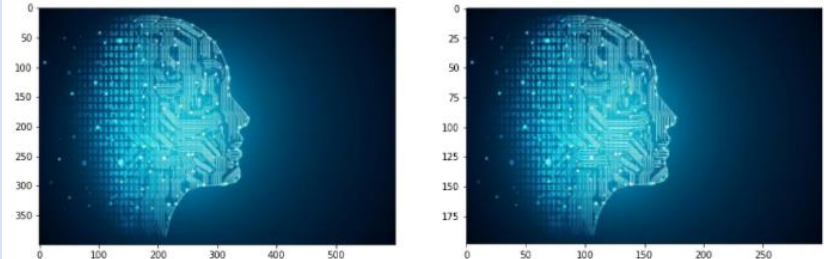
```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

```
plt.subplot(121)  
plt.imshow(gray, cmap='gray')  
plt.subplot(122)  
plt.imshow(hsv[...,:-1])  
plt.show()
```



```
import cv2  
import matplotlib.pyplot as plt  
plt.rcParams["figure.figsize"] = (15,15)  
  
imgpath = "./images/ai_image.jpg"  
  
img = cv2.imread(imgpath)  
# scaledimg = cv2.resize(img, dsize=(0, 0), fx=0.5, fy=0.5)  
h, w, ch = img.shape  
print("img size : ", img.shape)  
new_size=(int(w*0.5), int(h*0.5))  
scaledimg = cv2.resize(img, dsize=new_size, fx=0.0, fy=0.0, interpolation=cv2.INTER_CUBIC)  
print("scaled img size : ", scaledimg.shape)  
  
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)  
  
plt.subplot(121)  
plt.imshow(img[...,:-1])  
plt.subplot(122)  
plt.imshow(scaledimg[...,:-1])  
plt.show()
```

```
img size : (399, 600, 3)  
scaled img size : (199, 300, 3)
```



9.2 딥러닝 실습을 위한 사전 연습

영상처리

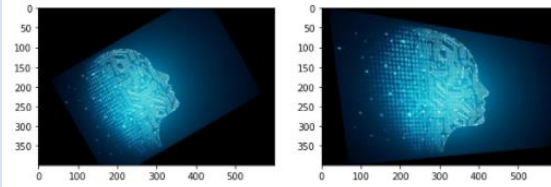
- Image Crop
`crop = img[y1:y2, x1:x2, :]`
- Image Flip
`img_vflip = cv2.flip(img, 0)`
- Image Filter
`mfilter = cv2.medianBlur(img, 5)`
- Image Warping
`dst = cv2.warpAffine(img, M, dsize)`

```
import numpy as np

center = (w//2, h//2)
angle = 30
scale = 0.75
dsize=(w, h)
rotmat = cv2.getRotationMatrix2D(center, angle, scale)
affine_dst = cv2.warpAffine(img, rotmat, dsize)

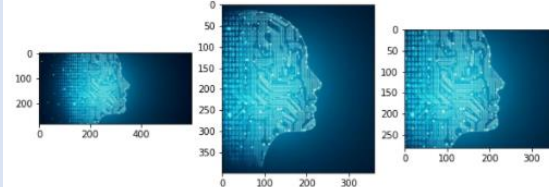
srcpts = np.array([[5, 5], [w-5, 5], [w-5, h-5], [5, h-5]], dtype=np.float32)
dstpts = np.array([[25, 5], [w-5, 100], [w-15, h-50], [75, h-5]], dtype=np.float32)
pmat = cv2.getPerspectiveTransform(srcpts, dstpts)
perspect_dst = cv2.warpPerspective(img, pmat, dsize)

plt.subplot(121)
plt.imshow(affine_dst[...,:-1])
plt.subplot(122)
plt.imshow(perspect_dst[...,:-1])
plt.show()
```



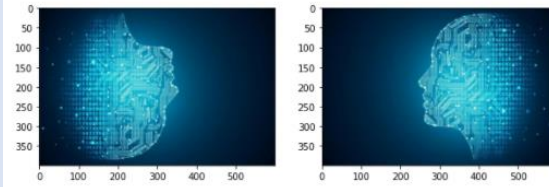
```
crop = int(h*0.15)
cropw = int(w*0.20)
imgcrop = img[crop:h-crop, :, :]
imgcropx = img[:, cropw:w-cropw, :]
imgcropxy = img[crop:h-crop, cropw:w-cropw, :]

plt.subplot(131)
plt.imshow(imgcrop[...,:-1])
plt.subplot(132)
plt.imshow(imgcropx[...,:-1])
plt.subplot(133)
plt.imshow(imgcropxy[...,:-1])
plt.show()
```



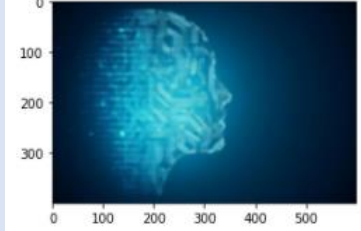
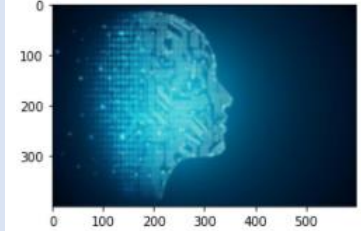
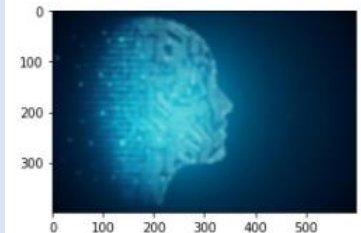
```
img_vflip = cv2.flip(img, 0)
img_hflip = cv2.flip(img, 1)

plt.subplot(121)
plt.imshow(img_vflip[...,:-1])
plt.subplot(122)
plt.imshow(img_hflip[...,:-1])
plt.show()
```



```
ksize=(9, 9)
abblur = cv2.blur(img, ksize)
gblur = cv2.GaussianBlur(img, ksize, 0)
mblur = cv2.medianBlur(img, ksize[0])

plt.subplot(311)
plt.imshow(abblur[...,:-1])
plt.subplot(312)
plt.imshow(gblur[...,:-1])
plt.subplot(313)
plt.imshow(mblur[...,:-1])
plt.show()
```



10.1 딥러닝 실습

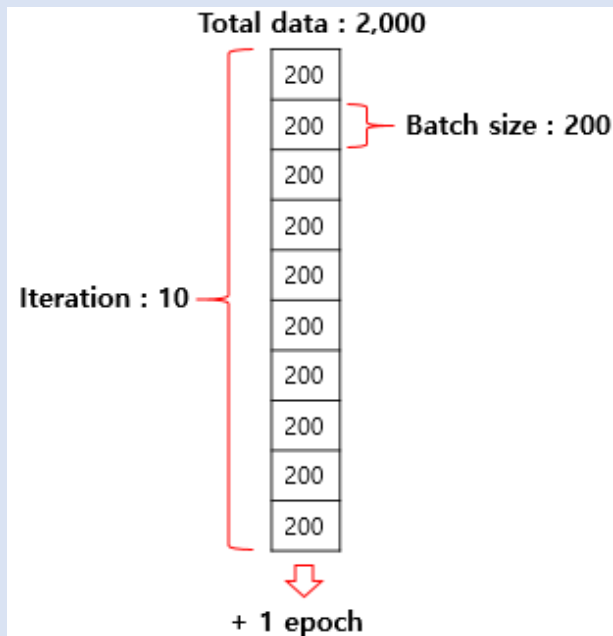
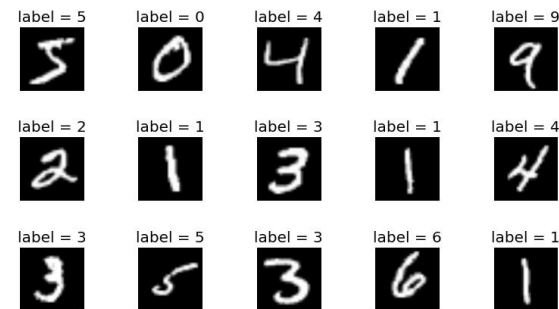
Data Handling

- MNIST : 손글씨 숫자 데이터 다운로드

```
mnist_train = datasets.MNIST(root='MNIST_data/', # 다운로드 경로 지정  
                             train=True, # True를 지정하면 훈련 데이터로 다운로드  
                             transform=transforms.ToTensor(), # 텐서로 변환  
                             download=True)  
  
mnist_test = datasets.MNIST(root='MNIST_data/', # 다운로드 경로 지정  
                             train=False, # False를 지정하면 테스트 데이터로 다운로드  
                             transform=transforms.ToTensor(), # 텐서로 변환  
                             download=True)
```
- 데이터 로더 설정

```
data_loader = torch.utils.data.DataLoader(dataset=mnist_train,  
                                           batch_size=batch_size,  
                                           shuffle=True,  
                                           drop_last=True)
```
- 데이터 로더 사용

```
for X, Y in data_loader:  
    X : (batch, ch, w, h) - 입력되는 이미지 데이터  
    Y : (batch, gt) - 입력되는 이미지 데이터에 대응되는 정답 레이블
```



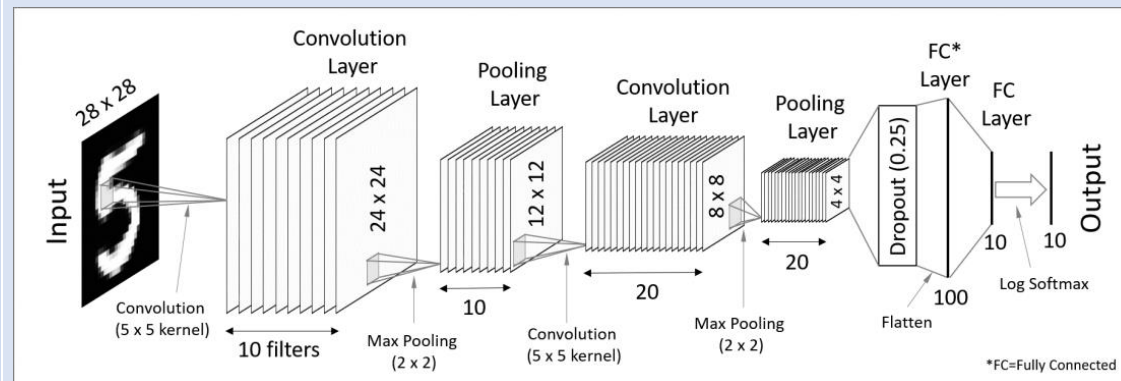
10.2 딥러닝 실습

Model 설계

```
# 첫번째 층
# ImgIn shape=(?, 28, 28, 1)
# Conv -> (?, 28, 28, 10)
# Pool -> (?, 15, 15, 10)
self.layer1 = torch.nn.Sequential(
    torch.nn.Conv2d(1, 10, kernel_size=5, stride=1, padding=3),
    torch.nn.ReLU(),
    torch.nn.MaxPool2d(kernel_size=2, stride=2))

# 두번째 층
# ImgIn shape=(?, 15, 15, 10)
# Conv -> (?, 15, 15, 20)
# Pool -> (?, 8, 8, 20)
self.layer2 = torch.nn.Sequential(
    torch.nn.Conv2d(10, 20, kernel_size=5, stride=1, padding=3),
    torch.nn.ReLU(),
    torch.nn.MaxPool2d(kernel_size=2, stride=2))

# 전결합층 8x8x20 inputs -> 10 outputs
self.fc = torch.nn.Linear(8 * 8 * 20, 10, bias=True)
```



10.3 딥러닝 실습

학습 코드

```
for epoch in range(training_epochs):
    avg_cost = 0

    for X, Y in data_loader: # 미니 배치 단위로 꺼내온다. X는 미니 배치, Y는 레이블.

        # image is already size of (28x28), no reshape
        # label is not one-hot encoded
        X = X.to(device)
        Y = Y.to(device)

        optimizer.zero_grad()

        hypothesis = model(X)

        cost = criterion(hypothesis, Y)

        cost.backward()
        optimizer.step()

        avg_cost += cost / total_batch

    print('[Epoch: {:>4}] cost = {:>.9}'.format(epoch + 1, avg_cost))
```

10.4 딥러닝 실습

테스트 코드

```
# 학습을 진행하지 않을 것이므로 torch.no_grad()
with torch.no_grad():
    X_test = mnist_test.test_data.view(len(mnist_test), 1, 28, 28).float().to(device)
    Y_test = mnist_test.test_labels.to(device)

    prediction = model(X_test)

    correct_prediction = torch.argmax(prediction, 1) == Y_test

    accuracy = correct_prediction.float().mean()
    print('Accuracy:', accuracy.item())
```

11. 현업의 딥러닝 실제 적용 분야와 한계

딥러닝 적용 분야와 한계는 무엇인가?

딥러닝이 실제 적용되는 분야들의 특징은 주로 자동화와 연관되어 있다.

이런 자동화를 처리하기 위해서는 지도학습을 주로 이용할 수 밖에 없다.

그러기 위해서는 많은 데이터가 존재해야 하며, 데이터의 레이블링 또한 수행 되어야 한다.

하지만 대부분의 자동화가 필요한 분야들은 데이터는 방대하지만, 레이블링이 수행되지 않은 데이터가 대부분이다.

추가로 개인정보가 포함된 데이터들은 외부로 반출이 어려워 실질적인 개발을 위해서는 개인정보를 마스킹하는 기술도 수반되어야 한다.

또한 GPU 컴퓨팅이 반드시 필요하다.

빠른 딥러닝 학습을 위해서는 GPU 컴퓨팅이 필요한데, 현재 많은 수요에 비해 공급이 부족한 것이 현실이다.

끝으로 학습한 모델을 실제 업무에 적용하기 위해서는 충분한 컴퓨팅 파워를 요구하는데,

현실적으로 고사양의 서버나 GPU 서버는 많은 비용이 든다.

따라서 최적화나 경량화를 통해 비교적 낮은 비용으로 모델을 동작하게 하는 기반 기술이 필요하다.

다만 업계에서는 아직 이런 코어 기술을 갖춘 엔지니어가 부족하다.

대부분 딥러닝의 기초를 배우고 사회에 발을 내딛는 사람들이라 실제 필요한 인재가 드물다.

12. 딥러닝의 미래

딥러닝의 미래는 어떤 모습으로 발전할 것인가?

- **초고성능의 컴퓨팅 출현**
 - 양자컴퓨터, 3D 메모리, 빠르고 방대한 양의 학습 처리 가능
- **융합 학습 플랫폼 출현**
 - 영상/음성/텍스트 등이 융합된 데이터 처리 및 학습이 가능한 플랫폼 개발로 인간의 교육과 유사한 형태로 발전
- **자가 학습 분야의 발전**
 - Self Learning 분야가 두드러지게 발전하고 새로운 형태의 자가학습 기술이 지속적으로 개발
- **산업으로의 확장**
 - 현재 일부 분야에서 딥러닝을 적용하여 업무 효율화를 이루고 있지만, 부분적이고 제한적임, 향후 산업 전분야로 확장
- **신규 분야의 출현**
 - 로봇 공학, 자율 주행 등에 이어 우주산업과 식량 산업, 유전자 분석, 신약 개발 등에서 새롭게 딥러닝을 적용하고 있음

13. 딥러닝 업무를 위한 학생 지도 방안

학생들이 향후 딥러닝 분야의 업무를 하고 싶어하는 경우 올바른 지도 방안은 무엇인가?

현업에서 실제 딥러닝 분야의 업무가 구체적으로 어떻게 이루어지고 있는지 정확한 정보 전달

그들의 커리어패스가 어떻게 만들어 졌고, 그들은 왜/어떻게 딥러닝 분야의 업무를 하고 있는지 파악하여 내용 공유

딥러닝 분야의 업무가 미래에 어떻게 변화되고 확장될 것인지 예측하고, 방향성을 설정하여 충분한 준비를 할 수 있도록 정보 전달

해당 분야에서 경쟁력 있는 업무는 무엇이고, 그렇게 되기 위해서는 어떤 교육과 경험을 해야 하는지 내용 파악 및 공유

14. Q&A

Q&A

End of Document