

# 대학혁신지원사업 3차년도 연구개발 결과보고서

.

세부프로그램번호		
세부프로그램명		
연구과제명		
연구책임자	소속:	직위: 성명: (인)
연구참여자 *없는 경우 항목 삭제	연구보조원	명

# 연구개발 결과보고서

		세부프로그램번호	
세부프로그램명			
연구과제명			
연구 기간	년	월	일 ~ 년 월 일
연구책임자	소 속	성 명	직 급
연구참여자	공동연구원	명	
	연구보조원	명	

2021년도 [대학혁신지원사업] 연구 과제를 성실히 수행하고 그에 대한 결과를 다음과 같이 보고합니다.

년
월
일

연구책임자 :
(인)

공동연구원 :
(인)

공동연구원 :
(인)

공동연구원 :
(인)

공동연구원 :
(인)

공동연구원 :
(인)

대학혁신지원사업 운영위원회 위원장(대학혁신지원사업단장) 귀하

# 목 차

## I . 서론

1. 연구의 필요성 및 목적
2. 연구 방법

## II . 인공지능(AI)

1. 인공지능(AI)
2. 인공지능 응용 분야
3. 인공지능(AI), 머신러닝(ML), 딥러닝(DL)

## III. 머신러닝과 딥러닝

1. 머신러닝(ML, Machine Learning)
2. 딥러닝(DL, Deep Learning)
3. 머신러닝(ML)과 딥러닝(DL)

## IV. GAN(Generative Adversarial Networks)

1. GAN(Generative Adversarial Networks)
2. GAN의 개념
3. GAN 모델 구조
4. GAN 종류
5. 응용 분야와 적용 사례
6. 악용사례

## V. TF2-GAN

## VI. Toonify Yourself(Toonification GAN)

## VII. 결론

## VIII. 참고 문헌

# I. 서론

## 1. 연구의 필요성 및 목적

### 1.1 연구 배경

- ① 최근 4차 산업 시대에 인공지능 관련 수요가 폭발적으로 증가
  - : 구글 딥마인드가 개발한 인공지능 바둑 프로그램인 알파고는 2016년 세계 최상 위급 프로기사인 이세돌 9단과의 5번의 경기에서 4승 1패로 승리하였다. 알파고의 학습, 즉 인공지능의 학습에서는 알고리즘과 데이터, 그리고 연산 능력이 중요하였으므로 심층 신경망(DNN)과 몬테카를로 트리 탐색(MCTS)를 사용하여 선택지 중 가장 유리한 선택을 하도록 학습되었다.
- ② 인공지능의 나아갈 방향인 비지도 학습에 관한 관심이 증가함에 따라 비지도 학습의 종류 중 하나인 GAN의 관심도 증가
  - : 인공지능의 목표는 미래를 예측하는 것이다. 지도학습은 정해진 정답이 제공되어야 하지만 비지도 학습은 정답이 제공하지 않아도 되기 때문에 비지도 학습을 사용하는 GAN에 대한 관심도가 증가한다.
- ③ 이미지 처리를 위한 다양한 딥러닝 기술들을 분석하고, 이미지의 특성을 도출할 수 있는 딥러닝 모델 연구 필요
- ④ 도출된 특성을 바탕으로 GAN 기술을 활용하여 이미지의 특성을 목적에 맞게 변형시킬 수 있는 필터링 기법의 연구 필요

### 1.2. 연구 목적 및 필요성

#### 1.2.1. 여러 플랫폼에서 인공지능(AI) 필터의 수요 증가

- ① instagram
  - : AI 필터를 제공함으로써 많은 사람이 관심을 가지고 사용함
- ② 저화질이거나 눈을 감은 사진 등의 이미지들을 간단한 작업을 통해 개선하고 싶어 하는 수요가 존재함
- ③ 상상만 해왔던 일을(ex. 웹툰 속 주인공 되어보기) 쉽게 이뤄낼 수 있음

#### 1.2.2. 다양한 캐릭터 제작에 대한 차별성 제공 가능

- ① GAN을 통해 본인과 비슷한 자신만의 캐릭터 제작 가능
- ② 자신과 비슷한 캐릭터를 사용하며 다른 메타버스 앱과의 차별성을 둘 수 있으며, 메타버스의 관심도와 사용량의 증가에 도움이 될 수 있음

### 1.2.3. 새로운 학습 방법론으로 부상

- ① 대표적인 비지도 학습 기존의 학습보다 더 많은 종류의 문제해결 가능성을 시사함
- ② 데이터를 기반으로 미래를 예측하는 데에 있어서 비지도 학습의 효율성이 유망함

## 2. 연구 방법

### 2.1. 사전 조사

#### 2.1.1. 인공지능 조사

- ① 인공지능의 개념, 역사, 인공지능의 중요성 및 응용 분야 조사
- ② 인공지능과 머신러닝, 딥러닝의 차이점 조사

#### 2.1.2. 머신러닝과 딥러닝 조사

- ① 머신러닝과 딥러닝 개념 조사
- ② 지도학습과 비지도 학습의 개념과 차이점 조사
- ③ 머신러닝과 딥러닝의 차이점 조사

#### 2.1.3. GAN 조사

- ① 다양한 종류의 GAN 조사 및 분석

### 2.2. StyleGAN2를 이용하여 사람 얼굴 디즈니화

- ① Toonify Yourself 기술 연구
- ② 프로그램 결괏값 도출

## Ⅱ. 인공지능(AI)

### 1. 인공지능(AI)

인공지능(AI, Artificial Intelligence)은 사고나 학습, 문제해결 등 인간이 가진 지적인 능력을 컴퓨팅 환경을 통하여 구현하는 기술이다. 즉, 컴퓨터가 사람처럼 사고를 할 수 있도록 인간의 사고 능력을 기계적으로 구현 및 자동화한 시스템을 말한다.

1950년 Computing Machinery and Intelligence 논문에서의 “기계는 생각할 수

있는가?”라는 질문은 인공지능의 시작을 알렸다. 인공지능은 1956년에 도입되었으며, 이후 많은 전문가에 의해 데이터의 처리 수준이 증가하고, 저장 기능 또한 향상되었다. 현재에는 기계(컴퓨터)를 이용한 학습인 ‘머신러닝(ML, Machine Learning)’과 여러 비선형 변환 기법을 조합하여 높은 수준의 추상화를 시도하는 기계학습 알고리즘의 집합인 ‘딥러닝(DL, Deep Learning)’을 통해 인간의 사고방식을 기계(컴퓨터)에 가르치는 형태로 발전하고 있다.

## 1.1. 인공지능(AI)의 잠재적 목표와 정의

### 1.1.1. 인간적인 접근

#### ① 인간적 접근

- 인간처럼 생각하는 시스템
- 인간처럼 행동하는 시스템

#### ② 이상적인 접근

- 합리적으로 생각하는 시스템
- 합리적으로 행동하는 시스템

## 1.2. 인공지능의 역사

인공지능(AI)은 1965년에 처음 등장하여 데이터의 양적 증가, 첨단 알고리즘 등이 발전한 오늘날 매우 활발하게 연구되고 있는 분야이다.

1950년대, 초기 인공지능(AI)연구에서는 문제해결, 기호법 등의 주제를 탐구하였으며 1960년대로 접어들면서 해당 연구에 관심을 보이던 미국방부가 인간의 기본적인 추론 방식을 흉내 낼 수 있도록 컴퓨터를 훈련하기 시작하였다. 1970년 국방 고등 연구 기획국(DARPA)이 수행한 도로 지도화 프로젝트가 사례 중 하나로 소개된다. 국방 고등 연구 기획국은 Siri, Alexa, Cortana와 같은 인공지능이 개발되기 전, 2003년에 지능형 개인 비서를 개발하기도 하였다.

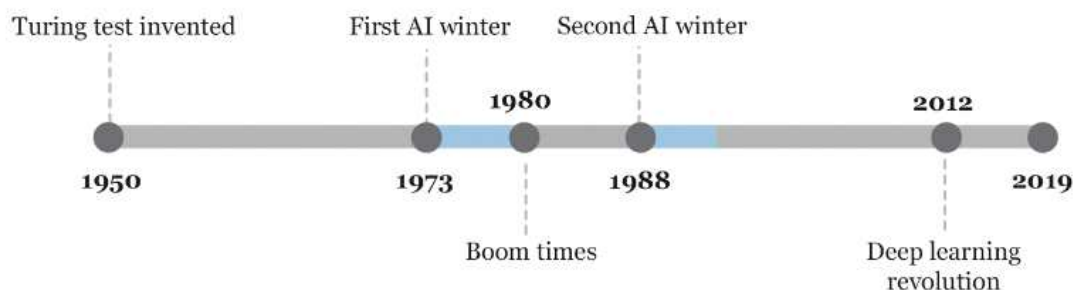


그림 01 | 인공지능의 역사

년도	연구	내용
1950년대 ~ 1970년대	신경망(Neural Networks)	신경망에 대한 초기 연구를 통해 '생각을 할 수 있는 기계'에 대한 기대감이 높아짐
1980년대 ~ 2010년대	머신러닝(Machine Learning)	머신러닝에 대한 관심이 높아짐
~ 현재	딥러닝(Deep Learning)	딥러닝으로 인공지능(AI) 시대가 본격화됨

### 1.2.1. 인공지능의 등장

1940년대 후반과 1950년대 초반에 이르러 수학, 철학, 공학, 경제 등의 분야의 과학자들에게서 인공적인 두뇌의 가능성이 논의되었으며, 인공지능(AI, Artificial Intelligence)은 1956년 미국 다트머스 대학에 있던 존 매카시 교수가 개최한 다트머스 회의를 통해 처음으로 용어가 사용되기 시작되었다. 당시 인공지능 연구의 핵심은 추론과 탐색이었다. 인간처럼 생각과 문제를 풀 수 있는 인공지능을 구현하려는 연구는 1970년대까지 활발하게 진행되었으나 복잡한 문제 풀이 수준까지는 도달할 수 없었다. 결국 인공지능과 관련된 연구는 급격한 빙하기를 맞게 되었다.

### 1.2.2. 스스로 학습하는 인공지능 : 빅데이터와 머신러닝

빙하기를 맞은 인공지능 관련 연구는 1980년대 다시 부상하기 시작했다. 이 시기에는 컴퓨터에 지식과 정보를 학습시키는 연구가 이루어졌으며, 여러 실용적인 전문가 시스템들이 개발되었다. 그러나 관리 방안에 대한 단점이 노출되었고, 이로 인해 1990년대 초까지 또 한 번의 빙하기를 맞게 되었다. 그러나 1990년대 후반, 인공지능 연구는 검색 엔진 등을 통해 과거와는 비교할 수 없을 정도로 방대한 데이터를 수집할 수 있는 인터넷이 도입되며 또 한 번 중흥기를 맞이하게 되었다. 이른바 머신러닝(Machine Learning)을 통해 많은 빅데이터를 분석하고, 이를 인공지능 시스템이 스스로 학습하는 형태로 진화하게 되었다.

### 1.2.3. 딥러닝(Deep Learning) 알고리즘

인간의 뇌를 모방한 신경망 네트워크(Neural Networks)로 이루어진 딥러닝(Deep Learning) 알고리즘은 기존의 머신러닝(Machine Learning)의 한계점을 뛰어넘을 수 있었다. 딥러닝은 2006년 캐나다 토론토 대학의 제프리 힌튼 교수가 처음 발표하였으며, 얀 레쿤과 앤드류 응과 같은 세계적인 딥러닝 크루 들에 의해 더욱 발전하게 되었고, 현재 구글과 페이스북, 바이두와 같은 글로벌 IT 회사에 영입되어 연구를 가속화 시키고 있다. 주로 영상과 이미지 이해, 음성인식, 기계번역 등에 쓰이는 딥러닝 알고리즘은 2012년 캐나다 토론토 대학의 알렉스 크리제브스키가 이미지넷(ImageNet)이라고 불리는 이미지 인식 경진 대회에서 딥러닝을 이용해 자체적으로 이미지를 인식해 내는 컴퓨터로 우승을 차지하며 또 한 번 획기적인 기술로 부상하게 되었다. 이는 딥러닝 연구에 GPU가 전면적으로 등장하게 된 계기가 되기도 하였다.

### 1.2.4. 딥러닝, GPU의 활용

이미지넷 경진대회는 1000개의 카테고리, 100만개의 이미지로 구성되며, 정확도를 겨루는 대회로, 10여년간 컴퓨터의 이미지 인식율은 75%를 넘기지 못했다. 그러나 알렉스는 나선형 신경망(CNN)을 활용해 알렉스넷(Alexnet)이라고 불리는 깊은 신경망(Deep Neural Network)을 설계하여 GPU를 활용해 많은 이미지 인식 훈련을 사용하는 방식을 사용하여 당시 84.7%라는 놀라운 정확도를 보여주었다. 이후 딥러닝 연구는 GPU와 함께 급속도로 발전하게 되었으며, 2015년에 개최한 이미지넷 경진대회에서는 마이크로소프트(MS)팀이 GPU를 활용해 96%가 넘는 정확도를 기록하였다.

### 1.2.5. GPU와 인공지능 알파고

구글 딥마인드의 알파고(AlphaGo)는 지도 학습과 비지도 학습을 동시에 사용하며, 비지도 학습의 한 종류인 강화학습의 기술로 학습을 진행하였다. 알파고는 16만 건이 넘는 프로기사의 기보로 매일 3만 번의 실전 경험을 쌓으며 스스로 학습하고, 성장하였다. 수많은 연산량은 176개의 GPU로 이루어진 고성능 시스템이 있었기에 가능했으며 이는 일반적인 CPU 시스템보다 30배 이상 연산속도가 빨랐으며, 이 때문에 짧은 시간에 효과적인 연산이 가능했고, 전력의 소모도 줄일 수 있었다. 알파고 개발 총책임자인 데이비드 실보 교수는 알파고의 브레인에 100개가 넘는 GPU가 있다고 할 정도로 알파고에서 GPU의 역할은 크게 자리 잡고 있다.



### 1.3. 인공지능(AI)의 중요성

#### 1.3.1. 반복적인 학습과 데이터의 발견을 자동화한다.

인공지능은 하드웨어 기반 로봇의 자동화와 달리, 수작업을 자동화하는 것을 넘어 반복적인 대량의 전산 작업을 간단하게 수행할 수 있다.

#### 1.3.2. 기존 제품에 지능을 더할 수 있다.

Apple의 Siri처럼 기존 제품 및 신제품에 새로운 기능으로 추가하여 제품의 개선을 이루어낼 수 있다. 이는 자동화, 대화 플랫폼, 스마트 머신 등의 제품이 대량의 데이터와 결합 되면서 많은 기술을 개선할 수 있다.

#### 1.3.3. 점진적인 학습 알고리즘으로 스스로를 개선한다.

인공지능은 데이터의 구조, 규칙성을 찾아내 알고리즘이 이를 학습하도록 지원한다. 새로운 데이터가 입력될 때마다 학습을 진행하고, 성능이 개선된다. 즉 데이터가 프로그래밍을 수행할 수 있도록 지원한다.

#### 1.3.4. 인공지능은 향상된 정확도를 제공한다.

Alexa, Google Search, Google Photos 등 많은 곳에서 모두 딥러닝 기술을 활용하고 있으며 이는 더 많은 사용자가 높은 정확도로 원하는 결과를 획득할 수 있도록 돕습니다. 현재 의료분야에서 딥러닝과 이미지 분류, 객체 인식 등의 인공지능 기술들을 MRI 이미지 분석에 사용하고 있으며, 이 기술은 방사선 기술자만큼이나 정확한 결과를 추출한다.

#### 1.3.5. 데이터의 활용도를 극대화한다.

알고리즘의 자체 러닝을 진행할 때, 모든 문제의 해결책은 데이터가 가지고 있어서 데이터 자체의 중요성이 높아진다. 따라서 최상의 데이터를 보유하는 것이 중요하게 된다.

## 2. 인공지능 응용 분야

오늘날, 대부분 업계에서는 인공지능(AI)의 적용과 활용을 요구하고 있으며 아래와 같은 산업에서 인공지능을 사용하고 있다.

### 3.1. 헬스케어

X-ray 판독과 맞춤 의료 서비스 제공에 인공지능(AI)을 활용할 수 있다. 해당 부분에서의 인공지능은 약을 먹을 시간을 알려주거나, 운동과 건강한 식습관 유지를 도와

주는 건강 관리 도우미의 역할을 수행할 수 있다.

### 3.2. 추천 엔진

인공지능(AI)은 과거 소비 행동 데이터를 사용해 효과적인 교차 판매 전략을 개발하는 데에 사용할 수 있는 데이터 추세를 발견하는 데 도움이 된다. 또한 재고 관리 시스템과 사이트의 레이아웃에도 인공지능 기술을 적용하여 더 나은 모습으로 개선할 수 있다. 예를 들어 온라인 챗봇은 인간 상담원을 대체하여 웹 사이트와 소셜 미디어 플랫폼 전반에 고객 참여에 대한 사고방식을 변화시키며 사용자를 위한 개인화된 제안을 제공한다.

### 3.3. 은행업

금융 기관에서 인공지능(AI) 기술을 사용해 사기일 가능성이 있는 정보를 식별하여 빠르고 정확하게 관리 작업을 자동화할 수 있다.

### 3.4. 스포츠

인공지능은 경기 이미지를 저장하고, 최적의 전략을 파악하여 경기를 보다 효과적으로 계획할 수 있도록 돕는 코치의 역할을 수행한다.

### 3.5. 음성인식

자동 음성 인식(ARS), 컴퓨터 음성인식 또는 음성 텍스트 변환이라고 부르기도 하며, 자연어 처리(NLP)를 사용해 사람의 음성을 서면 형식으로 변환하는 기능이다.

많은 모바일 장치에서 음성인식을 사용하여 음성 검색(예: Apple의 Siri)을 수행하여 문자 메시지에 대한 접근성을 향상한다.

### 3.3. 컴퓨터 비전

컴퓨터는 인공지능(AI) 기술을 통해 디지털 이미지, 비디오 및 기타 시각적 입력에서 의미 있는 정보를 도출하여 적절하게 활용할 수 있다.

### 3.4. 생활 속 인공지능

- ① 자율 주행 자동차 : 테슬라, 구글, 현대자동차, 네이버
- ② 스마트 스피커(AI) 비서 : 아마존, 구글, 바이두, KT, SK 텔레콤, 네이버 등
- ③ 챗봇 : 카카오톡 상담톡, 네이버 톡톡, 라인, 채널 톡 등
- ④ 인공지능 로봇 : 청소 로봇, 교육용 로봇, 운송 로봇, 동반자 로봇 등

- ⑤ 이미지 인식 : 페이스북, 구글, 마이크로소프트, 네이버
- ⑥ 개인화 추천 : 넷플릭스, 구글, 페이스북
- ⑦ 기계번역 : 구글, 네이버 파파고

#### 4. 인공지능(AI), 머신러닝(ML), 딥러닝(DL)

인공지능과 머신러닝, 딥러닝에는 범위에 차이가 있다. 인공지능의 범위가 가장 크며, 머신러닝이 중간이고, 딥러닝이 가장 작다. 즉, 인공지능은 일종의 컴퓨터 프로그램이며, 인공지능의 한 분야로 머신러닝과 딥러닝이 존재한다. 나아가 딥러닝은 머신러닝의 한 분야이다.

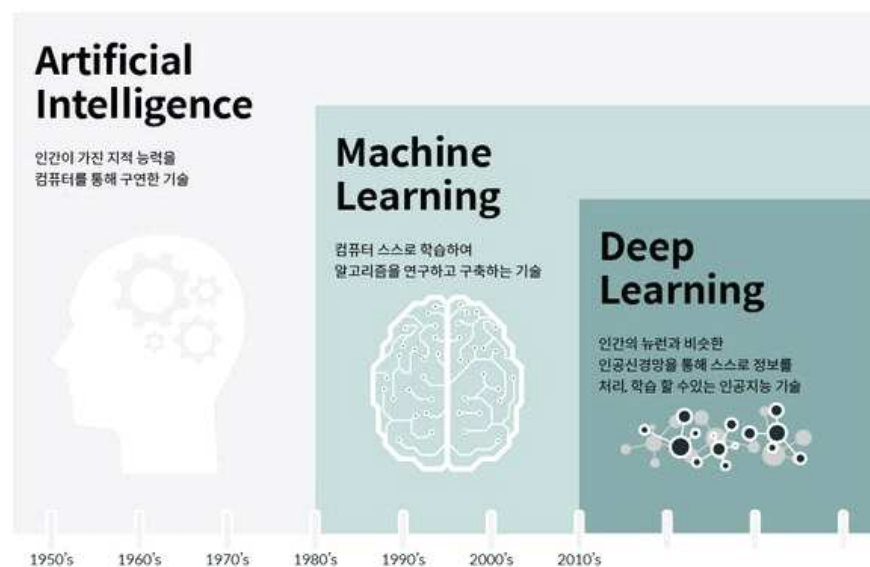


그림 02 | 인공지능(AI), 머신러닝(ML), 딥러닝(DL)의 관계

### Ⅲ. 머신러닝과 딥러닝

#### 1. 머신러닝(ML, Machine Learning)

인공지능을 소프트웨어적으로 구현하는 머신러닝(ML, Machine Learning)은 컴퓨터가 데이터를 학습하고, 스스로 패턴을 찾아내 적절한 작업을 수행하도록 하는 알고리즘이다. 즉, 기계가 코딩되지 않은 동작을 스스로 학습하여 수행하게 하는 연구 분야이다.

머신러닝은 미래의 출력을 예측할 수 있도록 알려진 입력과 출력 데이터를 기반으로 하는 지도 학습과 입력 데이터에서 숨겨진 패턴 혹은 고유 구조를 찾는 비지도 학습, 강화학습 세 가지로 분류된다.



그림 03 | 머신러닝의 분류

## 1.1. 지도학습(Supervised Learning)

지도학습이란, 증거를 기반으로 예측을 수행하는 학습을 말한다. 즉, 입력값과 출력값이 모두 존재하는 데이터(labeled data)를 학습하는 알고리즘을 말하며 학습을 통해 출력의 상관관계를 나타내는 함수를 유추한다. 여기서 함수는 인공신경망의 구조와 활성 함수에 따라 변경된다. 지도학습은 전통적인 인공신경망에서 주로 사용되는 기법으로 입력과 출력의 쌍이 필요하므로 데이터 구축에 드는 비용이 크다. 또한 지도학습은 최솟값을 찾는 최적화 문제로 귀결되나 최적화 문제의 차원이 매우 크기 때문에 초기의 조건에 대한 민감도가 높다.

대표적으로는 분류(Classification)와 회귀(Regression)를 사용하여 예측 모델을 개발한다.

### 1.1.1. 분류(Classification)

분류는 개념이나 주체를 인지, 차별화, 이해하는 과정을 말한다. 즉, 하나의 값이 아닌, 여러 가지 값 중 하나의 정답의 후보로 선택해야 하는 경우를 말한다.

두 개의 클래스로 분류하는 것은 이진 분류, 셋 이상의 여러 클래스로 분류하는 것을 다중 분류라고 한다.

#### ① 분류를 수행하기 위한 알고리즘

- 서포트 벡터 머신(SVM)
- 부스팅 및 배깅 결정 트리
- k-최근접 이웃
- 나이브 베이즈 분류
- 판별 분석
- 로지스틱 회귀
- 신경망

### 1.1.2. 회귀(Regression)

한 가지 값을 예측하는 방식의 인공지능을 회귀(Regression)라고 한다. 일반적으로 연속적인 숫자(벡터)를 예측하는 데에 사용한다. 회귀의 예시로는 몸무게를 예측하거나, 당뇨 지수를 예측하는 등의 수치를 맞추는 것을 회귀라고 부르기도 한다.

#### ① 회귀 알고리즘

- 선형 모델
- 비선형 모델
- 정규화
- 단계적 회귀 분석
- 부스팅 및 배깅 결정 트리
- 신경망
- 적응식 뉴로퍼지 학습

### 1.1.3. 분류와 회귀의 차이점

머신러닝이 맞춰야 하는 레이블(Label). 즉, 정답지가 이산적(Discrete)이라면 ‘분류’라고 하고, 정답지가 연속적(Continuous)이라면 ‘회귀’라고 한다.

이산적(Discrete)이란, 각각이 독립된 의미가 있고 연속적으로 표현하기 어려운 것을 말한다.

연속적(Continuous)이란, 정보의 연속성이 있는 것으로 사람의 키와 몸무게처럼 연속적으로 크기를 비교할 수 있는 것을 의미하고, 수학에서 실수에 해당한다.

## 1.2. 비지도 학습(Unsupervised Learning)

비지도 학습이란, 정답을 주지 않은 상태에서 학습하는 알고리즘으로, 컴퓨터가 스스로 데이터에 숨겨진 패턴이나 고유 구조를 찾는 것을 말한다. 즉, 입력값만 존재하는 데이터(unlabeled data)를 학습시키는 것으로 비슷한 특징을 모아 군집화하여 새로운 데이터에 관한 결과를 예측한다. 비지도 학습은 지도학습의 초기조건에 대한 민감도에 대한 문제를 해결하려는 방법으로 고안되었으며 입력값만으로도 학습할 수 있고, 이것은 특징이 비슷한 데이터의 분류를 수행하여 그 결과를 지도학습의 초기조건으로 활용할 수 있다. 입력과 출력값의 쌍을 구축할 필요가 없으므로 데이터의 구축에 대한 비용이 상대적으로 낮으며 출력값이 없으므로 데이터의 경향을 알 수는 있으나 정확한 의사결정은 어려울 수 있다.

비지도 학습은 데이터의 특징을 분석하여 군집화하므로 그룹핑 알고리즘이라고도 할 수 있으며, 최근 집중적으로 연구되고 있는 분야이다.

### 1.2.1. 군집화

#### ① 개념

군집화는 가장 일반적인 비지도 학습 기법이다. 이 기법은 탐색적인 데이터 분석을 통해 데이터에 숨겨진 패턴 혹은 그룹을 찾는 데에 사용된다.



그림 04 | 군집화

#### ② 응용 분야

- 유전자 서열 분석
- 시장 조사
- 객체 인식

### 1.3. 강화학습(Reinforcement Learning)

강화학습이란, 행동적인 머신러닝이며 옳은 행동을 했을 때 보상을 받으며 훈련을 시키는 것으로 어떤 환경에서 정의된 주체가 현재 상태(state)를 관찰하며 선택할 수 있는 행동(action) 중 가장 최대의 보상을 가져오는 행동이 무엇인지 학습하는 것이다. 강화학습은 정적인 데이터 셋에 의존하는 지도 학습이나 비지도 학습과는 다르게 역동적인 환경에서 동작하며 환경과의 상호작용으로 얻는 보상을 통해 학습한다는 차이점이 있다. 이러한 점은 지도 학습과 비지도 학습에서 필요한 훈련 전 데이터 수집 및 전처리, 레이블 지정 과정에 대한 필요성을 감소시킨다는 장점이 있으며 인간의 개입 없이 학습 행동을 자체적으로 시작할 수 있다는 것을 의미한다.

#### 1.3.1. 응용 분야

강화학습으로 훈련된 심층 신경망은 복잡한 행동을 표현할 수 있기에, 기존 방법으로는 해결하기 까다롭고, 다루기 어려운 응용 분야에 대안 방식으로 접근할 수 있다. 장기, 단기의 포상 사이 트레이드 오프가 존재하는 문제를 다루는 데에 적합하며, 이는 로봇 제어, 엘리베이터 스케줄링, 통신망, 백개먼과 체스 같은 게임에 성공적으로 적용되어 오고 있다.

#### 1.4. 지도학습과 비지도 학습

지도학습과 비지도 학습은 모두 하나의 데이터를 기반으로 미래를 예측하는 것에 목표를 두고 있다. 하지만 지도학습은 정답이 주어진 데이터만 사용해야 하므로 사용할 수 있는 데이터의 양에 한계가 있다. 또한 대량의 데이터에 대한 정답을 개발자가 직접 알려주어야 하는 정제 과정을 거쳐야 하므로 해당 과정에서 인간의 개입이 필요하다는 단점이 존재한다. 반면에 비지도 학습의 경우, 인간이 정답을 알려주지 않더라도 경쟁 과정에서 스스로 학습을 하며 생성 모델을 통해 직접 이미지나 음성을 만들어내기 때문에 지도학습형 알고리즘과 차별화가 있다. 따라서 인공지능 기술의 미래는 지도 학습이 아니라 비지도 학습이 선도하게 될 것이며, 이러한 비지도 학습의 선두주자로는 GAN이 있다.

#### 1.5. 머신러닝 라이브러리 : 사이킷런(Scikit-learn)

2007년 구글의 썸머 코드에서 처음 구현되었으며, 현재 가장 널리 사용되는 머신러닝 패키지 중 하나이다.

#### 1.6. 활용 사례

- ① 스마트폰 잠금 해제 기능
- ② 카메라 보정 추천 기능
- ③ 갤러리 그룹핑 기능

### 2. 딥러닝(DL, Deep Learning)

딥러닝(DL, Deep Learning)은 머신러닝의 일부분으로 인간의 뇌가 학습하는 방식을 모방한 인공신경망(Artificial Neural Network)의 구조를 가진다. 딥러닝은 여러 데이터를 통해 정보를 입력받아 실시간으로 분석하기에 인간의 개입 없이 분석과 물리적인 작업을 수행하며 자동화를 개선하는 인공지능과 서비스를 구현하게 된다.

#### 2.1. 딥러닝 동작 방식

딥러닝은 신경망 구조를 사용하여 동작하기 때문에 심층 신경망으로 불리기도 한다. 딥러닝 모델은 수동으로 특징을 추출하지 않고, 직접 기능을 학습하는 신경망 구조와 함께 레이블링 된 데이터를 활용하여 훈련된다.

#### 2.2. ANN(인공신경망 : Artificial Neural Network)

딥러닝은 인공신경망(Artificial Neural Network)를 기초로 하고 있으며 인공신경망이라고 불리는 ANN(Artificial Neural Network)은 사람의 신경망 원리와 구조를 모방해 만든 기계학습 알고리즘 중 하나이다. 이는 인간의 뇌에서 뉴런들이 신호, 자

극 등을 받아 해당 자극이 어떤 임계 값(threshold)을 넘어서게 되면 결과 신호를 전달하는 과정에서 착안한 것이다. 이 과정에서의 신호와 자극은 인공신경망에서 Input Data이며, 임계 값(threshold)은 가중치(weight), 자극에 의해 어떠한 행동을 하는 것은 Output 데이터에 비교할 수 있다.

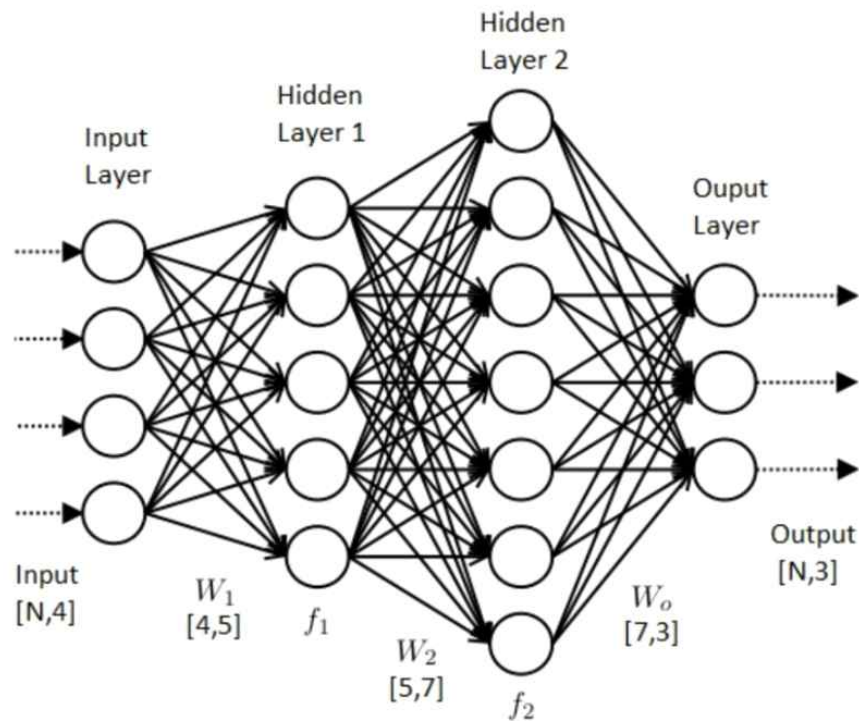


그림 05 | ANN(Artificial Neural Network) 인공신경망

### 2.2.1. ANN의 문제점

#### ① 학습 과정에서 파라미터의 최적값을 찾기 어려움

출력값을 결정하는 활성화 함수의 사용은 기울기의 값에 의해 가중치(weight)가 결정되는데, 이러한 gradient 값은 뒤로 갈수록 점점 작아져 결국 0에 수렴하는 오류를 낳기도 하며, 부분적인 에러를 최저 에러로 인식해 더는 학습을 진행하지 않는 경우가 발생하기도 한다.

#### ② 학습 시간의 속도가 느림

은닉층이 많게 된다면 학습하는 데에는 정확도가 올라가나, 그만큼 연산량이 증가하므로 이는 기하급수적으로 늘어나게 된다. 따라서 학습 시간이 오래 걸린다는 문제점이 있다.

그러나 최근 그래픽 카드의 발전으로 많은 연산량을 감당할 수 있을 정도로 하드웨어의 성능이 좋아져 점차 해결되어나가고 있다.



### ③ Overfitting에 따른 문제

오버피팅(Overfitting)에 대한 문제는 사전 훈련을 통해 방지할 수 있게 되었다.

## 2.3. DNN(Deep Neural Network)

DNN(Deep Neural Network)는 ANN의 여러 가지 문제가 해결되면서 모델 안에 은닉층의 개수를 많이 늘려 학습의 결과를 향상하는 방법이다. DNN은 은닉층을 2개 이상 가지고 있는 학습 방법을 뜻하며 컴퓨터가 스스로 분류 레이블(Label)을 만들어 내고, 공간을 왜곡하며 데이터를 구분 짓는 과정을 반복해 나가면서 최적의 구별선을 도출해낸다. 많은 데이터와 반복 학습, 사전학습, 오류 역전파 기법을 이용하여 현재 널리 사용되고 있다.

DNN을 응용해 만들어진 알고리즘의 예로는 CNN, RNN 등이 있으며 이 외에도 LSTM, GRU 등이 있다.

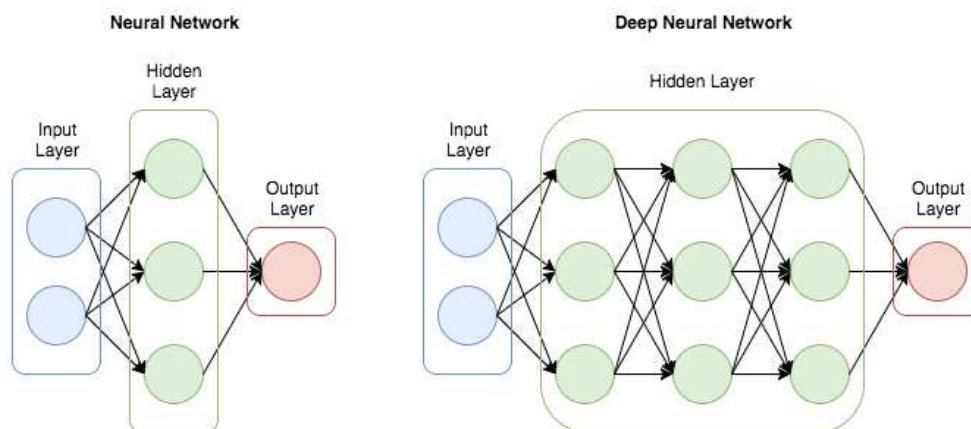


그림 06 | NN(Neural Network) vs DNN(Deep Neural Network)

## 2.4. CNN(합성곱신경망 : Convolutional Neural Networks)

CNN(Convolutional Neural Networks)은 가장 널리 사용되는 딥 신경망 중 하나이며, ConvNet라고 불리기도 한다.

CNN을 활용하면 수동으로 특징을 추출할 필요가 없기에 이미지를 분류하는 데에 사용되는 특징을 식별하지 않아도 된다. CNN은 이미지에서 특징을 직접 추출하여 작동되며, 관련 특징들은 사전에 훈련되지 않고, 신경망이 이미지 모음에서 훈련을 하는 동시에 학습이 된다. 이 알고리즘은 Convolution 과정과 Pooling 과정을 통해 진행되며, Convolution Layer와 Pooling Layer를 복합적으로 구성하여 알고리즘을 만든다.

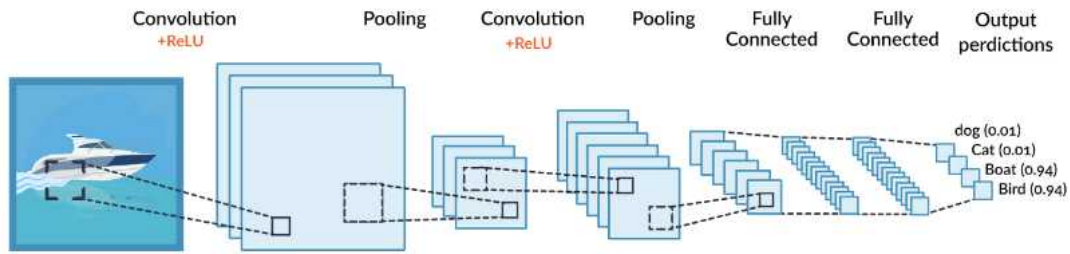


그림 07 | CNN(Convolutional Neural Network)

#### 2.4.1. Convolution 과정

데이터의 특징을 추출하는 과정이다. 데이터에 각 성분의 인접한 성분들을 조사하여 특징을 파악하고, 파악한 특징을 한 장으로 도출시킨다. 이때 도출된 장을 Convolution Layer라고 하며 이 과정은 하나의 압축과정으로 표현하고, 파라미터의 개수를 효과적으로 줄여주는 역할을 한다.

#### 2.4.2. Pooling 과정

Convolution 과정을 거친 레이어(Layer)의 사이즈를 줄여주는 과정이다. 단순히 데이터의 크기를 줄여주며 노이즈(Noise)를 상쇄시키고, 미세한 부분에서 일관적인 특징을 제공한다.

#### 2.4.3. 활용 분야

- ① 정보 추출
- ② 문장분류
- ③ 얼굴 인식

### 2.5. RNN(순환신경망 : Recurrent Neural Network)

RNN(Recurrent Neural Network) 알고리즘은 반복적이고, 순차적인 데이터(Sequential data) 학습에 최적화된 인공지능망의 한 종류이다. 내부에 순환 구조가 들어있다는 특징이 있으며, 이를 이용해 과거 학습의 가중치(weight)를 이용해 현재 학습에 반영한다. 기존의 지속적, 반복적, 순차적인 데이터 학습의 한계점을 해결한 알고리즘으로 현재의 학습과 과거 학습의 연결을 가능케 하고, 시간에 종속된다는 특징을 가지고 있다. RNN은 음성 웨이브 폼을 파악하거나 텍스트의 앞과 뒤 성분을 파악할 때 주로 사용한다.

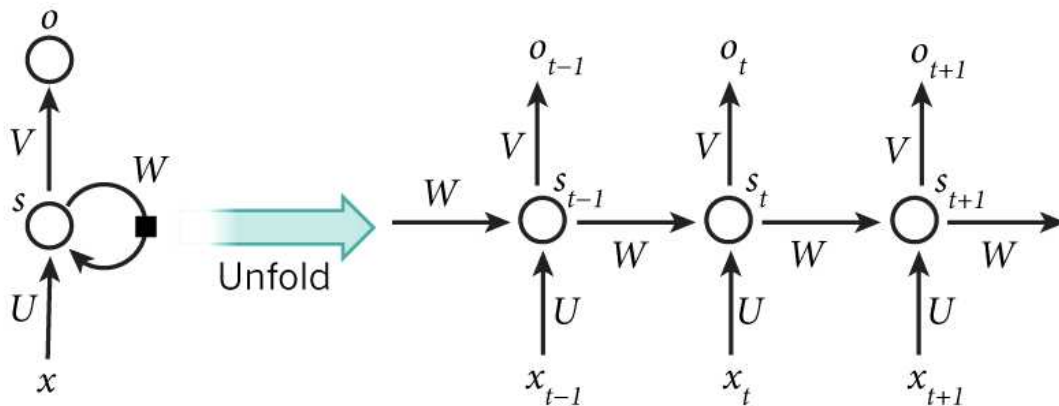


그림 08 | RNN(Recurrent Neural Network)

### 3.4.1. RNN의 문제점

단순한 형태의 RNN의 경우 역전파(backpropagation) 알고리즘을 기반으로 오랜 시간에 걸쳐 경향성이 나타난다. 그러나 데이터를 학습할 때, gradient가 비정상적으로 감소하거나, 증가하는 vanishing/exploding gradient problem이 발생한다는 문제점이 있다.

## 3.5. LSTM(Long Short Term Memory)

LSTM(Long Short Term Memory)은 기존의 RNN이 출력과 먼 위치에 있는 정보를 기억할 수 없다는 단점을 보완하기 위해 장기 및 단기 기억을 가능하게 설계한 신경망 구조를 말한다.

## 3.6. 응용 사례

### 2.3.1. 구글의 알파고

알파고에 많은 양의 바둑 상황 데이터를 입력으로 넣으면 알파고가 상황에 맞는 적절한 판단을 스스로 내린다. 알파고는 가능한 모든 상황에 따라 이길 수 있는 확률을 한발 앞서 계산하고, 승리 확률이 높은 확률의 수를 선택한다.

### 2.3.2. 무인 주행 자동차

과거에는 신호가 바뀌었는지, 차가 있는지 없는지 등의 질문을 모두 하나씩 입력으로 넣어주어야 했으나, 현재는 딥러닝을 활용해 발전하고 있다. 위험 상황과 정상적인 상황을 포함한 수많은 데이터를 넣어둘 경우, 컴퓨터가 자동으로 해당 데이터를 학습하여 자동으로 상황을 판단하게 된다.

### 2.3.3. 페이스북

페이스북에서 얼굴 사진을 올릴 경우, 자동으로 해당 이름이 태그되는 딥러닝 기술을 사용하고 있다. 이는 얼굴 인식 및 분류 하는 딥페이스 알고리즘을 활용하고 있는 것이며, 인식의 정확도는 인간이 인식하는 것과 거의 유사한 정확도를 보인다.

### 2.3.4. 네이버

네이버에서는 음성 검색 기능과 뉴스 요약, 이미지 분석에 딥러닝을 활용하고 있다.

## 2.4. 딥러닝 라이브러리 : 텐서플로(TensorFlow), 파이토치(Pytorch)

텐서플로(TensorFlow)	구글이 만든 딥러닝 라이브러리로, CPU / GPU를 사용해 인공신경망(Artificial Neural Network) 모델을 효율적으로 훈련하며 모델 구축과 서비스에 필요한 다양한 도구를 제공함. 텐서플로 2.0는 신경망 모델을 빠르게 구성할 수 있는 케라스(Keras)를 핵심 API로 채택하였으며, 이를 사용하면 간단한 모델에서 복잡한 모델까지 비교적 쉽게 만들 수 있음.
파이토치(Pytorch)	페이스북(Facebook)에서 개발하였으며 2016년에 공개한 파이썬 기반 오픈소스 머신러닝 라이브러리를 말함.
케라스(Keras)	다양한 인공지능 엔진에서 지원하며, 2015년에 공개된 파이썬 기반 오픈소스 신경망 라이브러리임. 텐서플로와 파이토치와 함께 널리 사용되고 있음.

## 4. 머신러닝(ML)과 딥러닝(DL)

### 4.1. 머신러닝과 딥러닝의 차이점

머신러닝과 딥러닝은 모두 인공지능의 하위 분야이며 그 중, 딥러닝은 머신러닝의 하위 분야로 존재한다. 이 둘의 차이점은 각각의 알고리즘이 어떻게 학습하는가에 있다. 아래 사진을 통해 머신러닝과 딥러닝의 차이점을 확인할 수 있다. 머신러닝(왼쪽 그림)은 수동으로 자동차의 특징과 분류기를 선택하여 이미지를 정렬하는 반면에 딥러닝(오른쪽 그림)을 사용하면 특징 추출과 모델링 단계가 자동으로 수행된다.

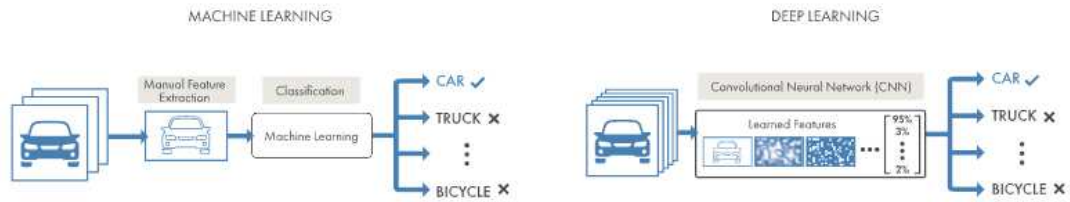


그림 09 | 머신러닝과 딥러닝 차이점 예제

	머신러닝(ML)	딥러닝(DL)
차이점	<p>주어진 데이터를 사람이 먼저 처리함. 사람이 먼저 컴퓨터에게 특정 패턴을 추출하는 방법을 지시하고, 이후 컴퓨터가 스스로 데이터의 특징을 분석하고 축적함. 이렇게 축적된 데이터로 컴퓨터는 문제를 해결함.</p> <p>ex. 엔지니어가 미리 데이터가 개인지 고양이인지 정의 내려야 함</p>	<p>머신러닝에서 사람이 하던 패턴 추출 작업이 생략됨. 컴퓨터가 스스로 데이터를 기반으로 학습을 할 수 있도록 정해진 신경망을 컴퓨터에 주고, 경험을 중심으로 학습을 수행함.</p> <p>ex. 개의 데이터와 개가 아닌 데이터들을 주면 자동으로 개인지 아닌지 군집화하고 분류함.</p>
공통점	인공지능과 관련된 서비스라는 것	

#### 4.3. 딥러닝과 강화학습의 차이점

	딥러닝	강화학습
차이점	<p>훈련 데이터의 학습을 기반으로 새로운 데이터에 적용.</p> <p>즉, 딥러닝은 데이터를 준비하는 단계가 필수적임.</p>	<p>훈련 데이터 없이 시행착오를 통해 보상을 최대화 하기 위해 지속해서 행동을 조정하는 동적 학습.</p> <p>즉, 주체(agent)와 환경 상호작용을 통해 학습하기에 데이터를 준비하는 단계가 환경 모델링으로 치환.</p>
공통점	자율적으로 학습 진행.	

## IV. GAN( Generative Advesarial Networks)

### 1. GAN(Generative Advesarial Networks)

GAN은 2014년 발표된 논문 Generative Adversarial Nets에서 처음 소개된 적대적 생성 신경망으로 인공지능망이 다양한 노이즈(Noise) 입력을 받아 기존에 존재하지 않은 새로운 진짜 같은 가짜 데이터를 만들어내는 AI 기술 중 하나이다. 기존의 이미지 처리에서 사용되는 딥러닝(Deep Learning)은 학습 데이터에 대해 여러 층으로 나뉜 하나의 인공지능망을 사용하여 학습시키는 방법을 사용하였으나, GAN은 2개의 인공지능망의 상호작용을 사용하여 진짜와 구분하기 힘들 정도로 정교한 가짜 이미지를 생성한다. 2개의 인공지능망은 각각 생성(Generator) 신경망과 판별(Discriminator) 신경망으로 구성되며, Goodfellow는 두 개의 신경망이 경쟁하는 관계에서 두 신경망 모델이 함께 성장할 수 있다고 주장하였다. 생성 신경망은 최대한 진짜 같은 화폐를 만들어 경찰을 속이는 위조 지폐범, 판별 신경망은 해당 화폐가 진짜 화폐인지 가짜 화폐인지 완벽하게 분류하여 위조 지폐범을 검거하는 경찰에 비유할 수 있다.



그림 10 | GAN 모델의 개념\_경찰과 위조 지폐범에 비유

### 2. 개념

생성적 적대 신경망 GAN의 2개의 인공지능망은 최대한 진짜 같은 데이터를 만들기 위한 학습을 진행하는 생성(Generative) 신경망과 생성 신경망이 만든 이미지가 진짜와 가짜를 판단하기 위한 학습을 진행하는 판별(Discriminative) 신경망으로 구분되며, 해당 신경망 모델을 적대적으로 학습시켜 실제 데이터와 비슷한 여러 가짜 데이터를 생성한다.

적대적인 관계에 있는 2개의 신경망을 학습시키는 것을 Adversarial training이라고 하는데, 이는 기본적으로 학습하고자 하는 데이터가 정해진 레이블(Label)이 존재하지 않기에 비지도 학습으로 분류된다.

생성 신경망은 판별 신경망을 속이지 못한 데이터를 입력으로 받고, 판별 신경망은 생성 신경망이 만들어낸 데이터 중 속은 데이터를 입력으로 받아 학습하며, 해당 과정의 반복을 통해 더욱 실제에 가까운 정교한 거짓 데이터를 만들게 된다.

### 3. 모델 구조

GAN은 크게 생성 모델(Generative model)과 판별모델(Discriminative model)이라고 부르는 두 개의 모델로 이루어져 있다. 두 모델은 모두 다층의 신경망(Multilayer perceptron)으로 이루어져 있으며, 생성 모델(G)은 임의의 노이즈(Noise)를 입력을 받아 실제와 비슷한 가짜 데이터를 만들어 내고, 판별모델(D)은 실제 데이터와 생성 모델이 만들어 낸 가짜 데이터를 입력받아 어떤 것이 진짜 데이터인지 판별한다.

#### 3.1. 생성 모델(G, Generative model, 생성기)

- 실제 데이터의 분포를 파악하여, 이와 비슷한 데이터를 만들어내도록 학습.
- 실제 데이터를 학습하고, 이를 바탕으로 거짓 데이터를 생성해 냄.
- 즉, 실제에 가까운 거짓인 데이터를 생성하는 것이 목적임.
- ex) 위조 지폐범 : 위조지폐를 만들어 들이지 않고 사용하는 것이 목적이며, 경찰을 속이지 못한 데이터에 대한 학습 진행.

#### 3.2. 판별모델(D, Discriminative model, 판별기)

- 실제 데이터와 생성기(G)가 생성한 가짜 모델을 구별하도록 학습.
- 생성자가 만든 데이터가 참인지 거짓인지 판별함.
- 즉, 생성자의 거짓 데이터에 놀아나지 않는 것이 목적임.
- ex) 경찰 : 지폐를 보고 진짜인지 가짜인지 판단하는 것이 목적이며, 위조지폐범에게 속은 데이터에 대한 학습 진행

#### 3.3. 학습

GAN의 학습 과정은 먼저 판별모델을 학습시킨 후, 생성 모델을 학습시키는 과정으로 진행된다.

분류 모델은 진짜 데이터를 진짜라고 분류할 수 있도록 학습을 시키고, 생성 모델이 생성한 데이터는 가짜로 분류할 수 있도록 학습시킨다. 해당 학습이 진행된 후, 학습이 진행된 분류 모델을 속이는 방향으로 생성 모델을 학습시키게 된다.

### 3.4. 목적 함수

#### 3.4.1 확률분포(Probability distribution)

##### ① 개념

확률분포(Probability distribution)란, 확률 변수가 특정한 값을 가질 확률을 나타내는 함수를 말한다.

##### ② GAN에서의 확률분포

GAN에서 다루고자 하는 모든 데이터는 확률분포를 가진 랜덤변수(Random Variable)이다. 그러므로 GAN을 정확하게 이해하기 위해서는 확률분포(Probability distribution)에 대해 알고 있어야 한다. 이를 수학적으로 표현하자면 생성(G) 신경망은 원 데이터의 확률분포를 알아내려 노력하고, 학습이 종료되면 원 데이터의 확률분포를 따르는 새로운 데이터를 만들어내게 된다.

랜덤변수(Random Variable)는 측정을 시행할 때마다 매번 다른 값이 나오게 된다. 그러나 생성되는 값은 확률분포를 따르는 데이터를 임의로 생성하기 때문에 해당 데이터는 확률분포를 구할 때 사용한 원 데이터와 유사한 값을 가지게 된다. 즉, GAN은 원 데이터와 확률분포를 정확하게 공유하는 무한히 새로운 데이터를 생성해낼 수 있음을 의미한다.

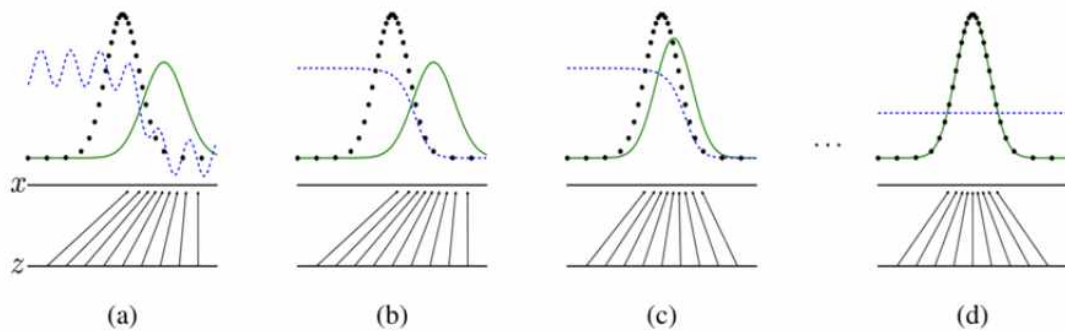


그림 11 | GAN에서 학습을 통해 확률분포를 맞춰 나가는 과정

검정선	원 데이터의 확률분포
녹색	GAN이 만들어내는 확률분포
파랑	판별자의 확률분포
x	실제 데이터의 정의역 일부



z	샘플링(Sampling)되는 분포의 영역
x에서 z로 가는 화살표	노이즈(Noise)를 생성 모델(G)에 넣어 샘플을 만드는 과정

위의 그래프를 통해 원 데이터의 확률분포가 학습이 진행됨에 따라서 GAN이 만들어내는 확률분포와 거의 같아지는 것을 확인할 수 있다. 그림 11의 (d)의 경우 더는 분류를 진행하더라도 의미가 없는 0.5의 확률값을 나타내게 된다. 즉, GAN을 통해 만들어진 데이터가 진짜인지 가짜인지 정확하게 분류할 확률이 0.5가 되면서 분류의 의미가 없어지게 되고, 생성자가 실제 데이터와 분류를 할 수 없을 정도로 유사한 데이터를 만들어 낼 수 있게 되었음을 의미한다.

### 3.4.2. MIN MAX problem

생성 모델(G)과 판별 모델(D)은 서로 경쟁하며 학습과 발전을 하게 되고, 어느 순간 진짜와 가짜를 구분할 수 없게 된다. 이때 생성 모델(G)을 학습시키기 위해서는 판별 모델(D)이 실수할 확률, 즉 생성 모델(G)이 만들었으나 실제 데이터라고 분류할 확률을 최대화하면 된다. GAN은 아래의 수식처럼 MIN MAX problem을 풀어나가는 방식으로 학습을 진행한다. 판별 신경망인 D는 손실 함수의 값을 최대화하고, 생성 신경망인 G는 손실 함수의 값을 최소화한다.

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

그림 12 | GAN의 손실함수 V(D, G)

$x \sim P_{data}(x)$	실제 데이터에 대한 확률분포에서 샘플링한 데이터
$z \sim P_z(z)$	정규 분포를 사용하는 임의의 노이즈에서 샘플링한 데이터
D(x)	분류(판별자)이며, x가 모델에 입력되었을 때, 진짜일 확률이 0과 1의 범위로 표현됨 (데이터가 진짜이면 1, 가짜이면 0의 값을 가짐)
G(z)	z라는 노이즈가 입력되면, 이를 바탕으로 생성 모델이 생성한 가짜 데이터

$D(G(z))$	생성자(G)가 만들어낸 데이터인 $G(z)$ 가 진짜이면 1, 가짜이면 0의 값을 가짐
-----------	--

① 판별자(D)가  $V(D, G)$ 를 최대화하는 관점

- 우변의 첫 번째 항과 두 번째 항이 모두 최대가 되어야 한다. 따라서  $\log(D(x))$ 와  $\log(1 - D(G(z)))$ 가 최대가 되어야 한다.
- 즉,  $D(x)$ 는 1이 되어야 한다. 이는 실제 데이터를 진짜라고 분류할 수 있도록  $D$ 를 학습하는 것을 의미한다.
- 즉,  $D(G(z))$ 는 0이 되어야 한다. 이는 생성자가 만들어낸 가짜 데이터를 가짜라고 분류할 수 있도록 학습하는 것을 의미한다.
- $\therefore V(D, G)$ 가 최대(Max)가 될 수 있도록 판별자(D)를 학습하는 것은 판별자(D)가 진짜 데이터를 진짜로, 가짜 데이터를 가짜로 분류할 수 있도록 학습을 하는 과정이다.

② 생성자(G)가  $V(D, G)$ 를 최소화하는 관점

- 우변의 첫 번째 항은  $G$ 가 포함되어 있지 않기 때문에 생략할 수 있고, 두 번째 항인  $\log(1 - D(G(z)))$ 가 최소가 되어야 한다.
- 즉,  $\log(1 - D(G(z)))$ 은 0이 되어야 한다. 이는  $D(G(z))$ 가 1이 되어야 함을 의미하며, 판별자가 진짜로 분류할 만큼 완벽에 가까운 가짜 데이터를 생성할 수 있도록 생성자를 학습시키는 것을 의미한다.

$\therefore$  손실 함수인  $V(D, G)$ 를 최대화하는 방향으로 판별자(D)를 학습시키고, 이를 최소화 하는 방향으로 생성자(G)를 학습시키는 것을 'MIN MAX problem'이라고 한다.

### 3.4. 한계

GAN은 대표적인 Generative model로 Image Generation 등에서 아주 좋은 성능을 보였다. 그러나 GAN은 아직 해결해야 하는 문제점들이 남아있다.

#### 3.4.1. Mode-collapse

GAN을 학습시키다 보면 생성 모델이 계속 비슷한 이미지만 생성하고, 다양한 이미지를 만들어내지 못하는 경우가 있는데, 이를 Mode-collapse라고 한다. 즉, 학습의 다양성이 떨어지는 것을 의미한다.

Mode는 최빈값. 즉, 가장 빈도수가 높은 것을 말하며 Mode-collapse를 MNIST 숫자 데이터(0 ~ 9)에서 설명하자면, Mode는 0부터 9까지 총 10개, 랜덤 노이즈(z)를 입력으로 받는 생성 모델이 판별모델을 속이기 위해 노이즈(Noise)를

변환하는데, 이렇게 변환된 데이터의 레이블(Label)이 특정한 숫자(Mode)에 치우치는 것을 Mode-collapse가 발생했다고 한다.

즉, Mode-collapse는 생성 모델이 판별모델을 속이기 위한 출력을 한 숫자에 가까운 벡터만 생성한다는 것이다.

### 3.4.2. 기타

Mode-collapse 외에도 기존의 딥러닝 이미지 처리 방식과는 달리 성능을 객관적인 수치로 표시하는 방법이 부족하다는 것과 4K와 같은 고화질 영상을 제작해 내기 위해서는 고성능의 메모리가 필요하기에 발전 속도가 느리다는 한계점이 존재한다.

## 4. GAN 종류

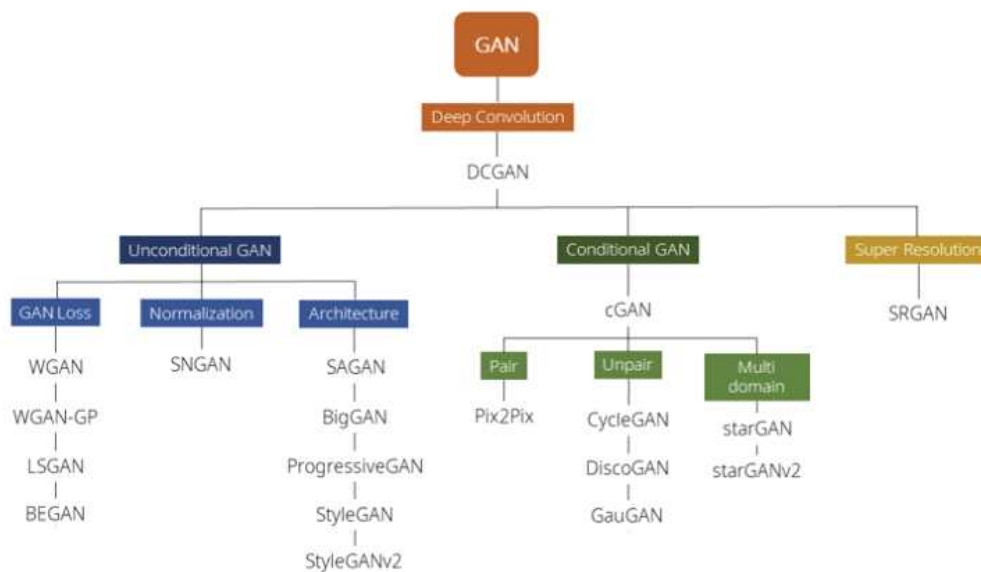


그림 13 | GAN의 구조 및 종류

### 4.1. DCGAN(Deep Convolutional GAN)

GAN은 생성자와 판별자가 서로 경쟁하며 학습을 하는 구조이기 때문에 학습이 불안정하다는 단점을 가지고 있다. 그러나 이러한 점을 극복하기 위해 DCGAN은 지도 학습인 CNN(Convolutional Neural Networks)을 비지도 학습인 GAN에 적용해 안정성을 증가시켰다. 이는 이미지의 벡터 산술 연산이 가능하고, 학습한 필터(Filter)들을 시각화할 수 있으며, 특정 물체를 생성해낼 수 있다.

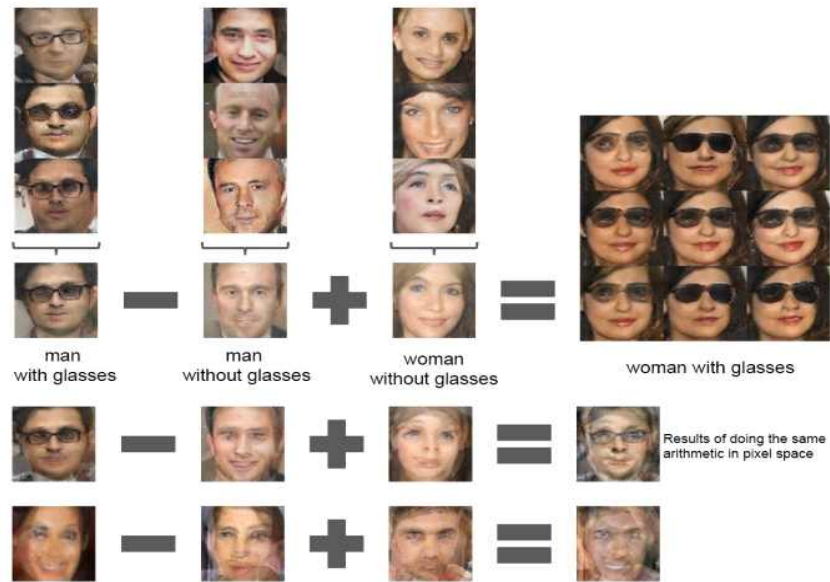


그림 14 | DCGAN의 이미지 벡터 산술 연산

#### 4.1.1. DCGAN에 적용된 CNN 구조

##### ① Strided Convolution 적용

DCGAN은 Strided Convolution을 convolution 과정 안에 포함해 진행하였다. 이때, Spatial Pooling은 down sample의 대표 기법의 하나로, 생성자(G)에는 Spatial Pooling을 진행하여 공간 해상도를 낮추는 Spatial downsampling 과정을 학습시켰으며, 판별자(D)는 Spatial upsampling 과정을 학습시켰다.

##### ② Fully-Connected Layer(FC Layer) 제거

Noise Vector의 Z에 존재하는 layer와 마지막 Output 결과를 판단하는 마지막 layer를 제외하고 FC layer를 전부 제외하였다.

##### ③ Black Box 문제해결

DCGAN에 판별자(Discriminator)를 학습시켜, 특정 부분이 활성화되는 필터를 발견하였고, 이를 통해 Black Box 문제를 해결할 수 있었다. 아래의 그림 15을 보면, 침대와 창문 등이 활성화되는 모습을 확인할 수 있다.

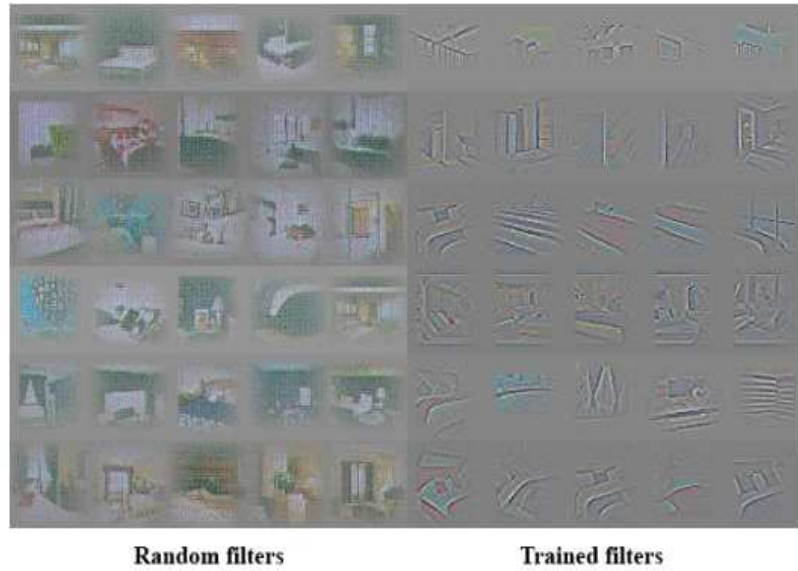


그림 15 | DCGAN의 활성화 필터

## 4.2. StyleGAN

StyleGAN은 PGGAN을 기반으로 한 GAN의 한 종류로, style transfer와 style mixing, stochastic variation을 사용한다. style mixing은 각 layer와 block이 특정 부분의 스타일에 대해서 구분할 수 있도록 진행하여, 2개의 latent code를 섞어 생성된 이미지를 나타내며, stochastic variation은 머릿결과 같이 인지에 영향을 끼치며, 랜덤하게 생성되는 요소를 확률적(stochastic)으로 여기는 것을 말한다.

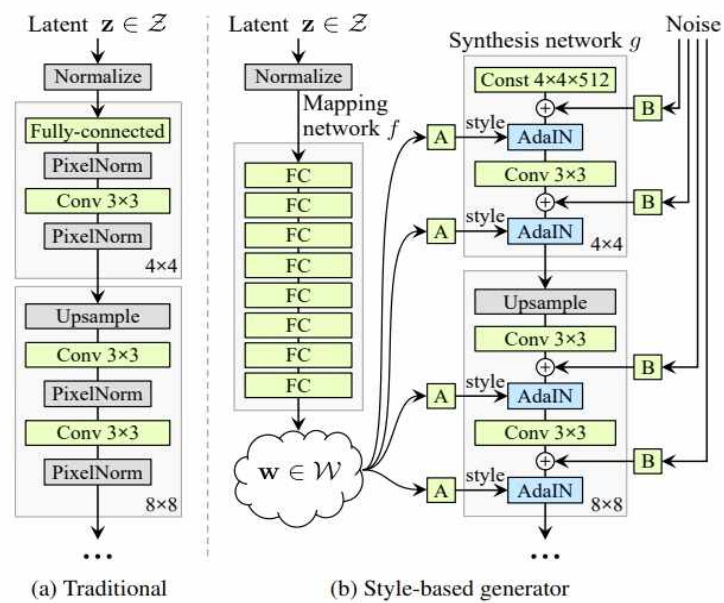


그림 16 | StyleGAN 생성자(Generoator)

#### 4.2.1. 특징

##### ① Progressive GAN

Progressive GAN은 저해상도의 이미지로 네트워크의 레이어(Layer)를 추가 하면서 점차 해상도를 높여가는 GAN이다. 즉, 모든 부분을 동시에 학습하는 것이 아니라 대략적인 이미지를 우선적으로 학습한 후에 세부적으로 학습해나간다.

##### ② Style mixing

각 이미지가 잘 융합되어 다른 레이어(Layer)에 관여하지 않도록 만드는 방법이며, 2개의 latent input vector( $z$ )를 입력한다. 인접한 레이어(Layer) 간의 Style 상관 관계를 줄이기 위해서 crossover point를 설정하여  $w_1$ 은 point 이전의 벡터로,  $w_2$ 는 point 이후의 벡터로 사용한다. 이때, 뒤에서 생성되는 이미 지일수록 convolution layer 측면에서 보면 patch 크기가 작아져 세밀한 변화가 가능해진다.

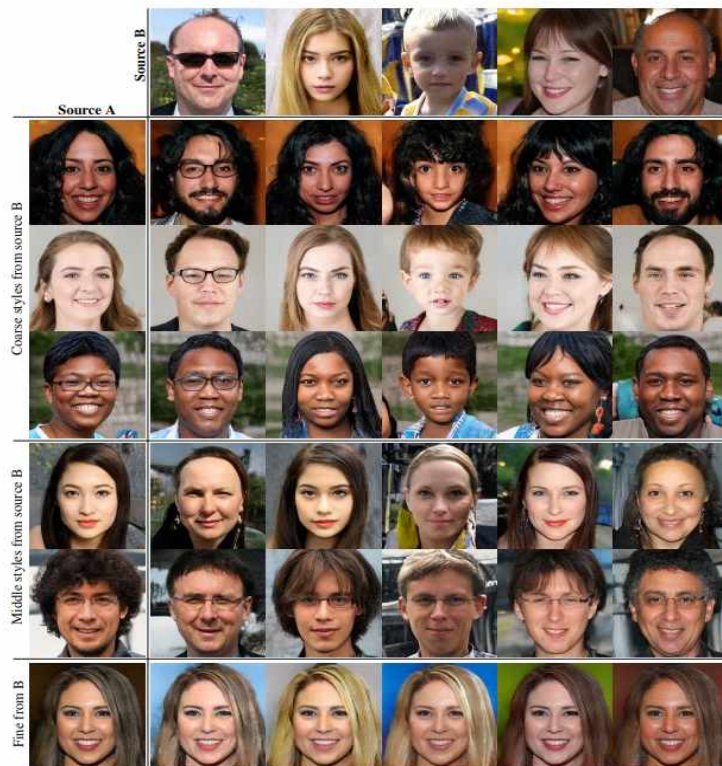


그림 17 | Style mixing

### 4.3. infoGAN

infoGAN에서는 엔트로피와 상호정보량(mutual information)이라는 정보이론에서 사용하는 개념을 GAN에 적용한다면 비지도 학습만으로 데이터의 특징을 적절하게 표현할 수 있다는 것을 주장한다. infoGAN은 데이터를 설명할 수 있는 변수들을 비지도 학습하며, 이 학습이 제대로 된다는 가정하에 학습한 변수로 데이터를 분류(classification)하는 것까지 가능해진다.

### 4.4. LSGAN(Least Squares GAN)

LSGAN(Least Squares GAN)은 손실 함수(Loss function)인 Sigmoid cross entropy loss가 gradient vanishing을 일으켜 학습이 정상적으로 진행되지 않아 생성한 이미지가 실제 이미지와 차이가 나는 것을 보완하기 위해 등장하였다. LSGAN은 least square을 loss function으로 적용했으며, decision boundary와 가까이 있는 샘플은 진짜로 구분하고, 멀리 떨어져 있는 샘플은 가짜로 구분하여 패널티를 주어 vanishing gradient를 방지한다.

#### 4.4.1. 기울기 소실(Gradient Vanishing)

기울기 소실은 활성화 함수의 기울기와 관련이 있다. Sigmoid 함수는 0과 1 사이의 값을 출력하는데, 이때  $x$ 의 값이 크거나 작아짐에 따라서 기울기의 값이 0으로 수렴하게 된다. 따라서 역 전파과정에서 미분 값이 곱해지면서 출력층과 멀어질수록 기울기의 값이 매우 작아지게 되며 결국 결괏값이 소실되어버리는 문제가 발생할 수 있다.

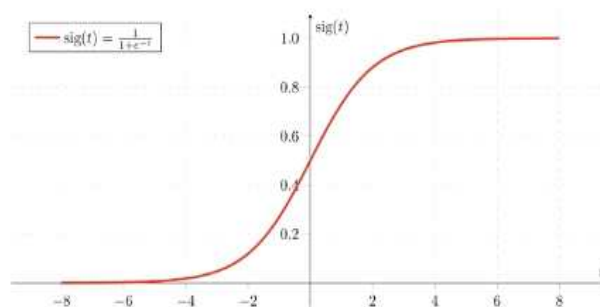


그림 18 | Sigmoid 함수

### 4.5. SGAN(Semi-Supervised with GAN)

SGAN(Semi-Supervised with GAN)은 semi-supervised learning을 하는 GAN으로 지도학습과 비지도 학습의 중간 부분인 준 지도학습의 GAN이다. 레이블(Label)



이 있는 지도학습의 데이터(Supervised data)와 레이블(Label)이 없는 비지도 학습의 데이터(Unsupervised data)를 결합하여 상호보완함으로써 더욱 효율적인 학습을 가능하게 한다. 기존의 GAN과 달리 시그모이드 함수(Sigmoid function)가 아닌 softmax 함수를 사용하며, DCGAN에 기본 구조를 두고 있다.

#### 4.6. Pix2PixGAN

Pix2PixGAN은 CGAN을 확장 시킨 GAN의 한 종류로 paired image-to-image translation을 추가한 것이다. 생성자(Generator)에 이미지가 들어가고, translated에 이미지가 출력되는 형식을 가진다. Pix2PixGAN의 최종 목표는 일반적인 GAN과 같이 진짜와 가짜를 구별하여 판별자(Discriminator)를 속이는 이미지를 생성해내는 것이다. 또한 UNET을 생성자(Generator)로 사용해 downsampling을 할 때 발생하는 손실정보를 보존하고, vanishing gradient를 방지한다.

#### 4.7. CycleGAN

CycleGAN은 특징이 겹치지 않은 서로 다른 이미지를 학습하는 GAN이다. Pix2PixGAN은 입력값  $x$ 와 출력값  $y$ 에 대해 유사성이 있는 이미지(paired)를 다루는 반면, CycleGAN은 유사성이 없는 이미지(unpaired)를 학습하여 순환 일관성 손실 함수(Cycle Consistency Loss Function)를 사용한다.

##### 4.7.1. 활용 사례

- ① 위성사진을 지도로 변환 가능 (반대로 지도를 위성사진으로 변환도 가능)
- ② 말과 얼룩말을 서로 변환할 수 있음
- ③ 그림의 계절이나 화풍도 변환할 수 있음.

### 5. GAN의 응용 분야와 적용 사례

GAN은 기존에 존재하지 않았던 새로운 이미지를 생성하고, 입력 이미지나 비디오를 다른 형태나 정보를 가진 이미지, 비디오로 변환해주기 때문에 최근 급속도로 발전하여 많은 관심을 받고 있다.

#### 5.1. 가짜 이미지 생성

GAN은 실제 이미지 학습을 통해 거짓 이미지를 만들어낸다. 이는 과거 전문가가 포토샵 등을 이용해 일일이 작업해야 했던 일들을 빠르고 쉽게 작업할 수 있도록 돕는다.



### 5.1.1. NVIDIA GAN

2017년 글로벌 GPU 설계 회사인 NVIDIA에서 공개한 ‘실존하지 않는 사람들의 이미지’가 GAN의 이미지 생성의 대표적인 적용 사례이다. NVIDIA는 기존 GAN의 결과들보다 이미지 품질과 안정성, 다양성 등을 향상한 새로운 훈련 방법론을 제시하였으며 그 결과 이미지는 사람의 눈으로는 실존 사람인지 가상의 사람인지 판별할 수 없는 수준이 되었다. NVIDIA가 제시한 GAN의 핵심은 생성자(G)와 판별자(D)를 모두 낮은 결과 값에서부터 훈련이 시작되면서 모델이 천천히 학습하도록 새로운 레이어를 쌓아가며 점진적으로 성장시키는 것이다. 또한 더 나아가 사람뿐만 아니라 침실, 소파, 버스, 화분과 같은 사물들을 실제와 같은 이미지로 만들어 낼 수 있음을 보여주었고, 이는 저해상도의 사진을 고해상도로 만들어주는 이미지 복원 기술에도 사용할 수 있다.



그림 19 | NVIDIA GAN 적용 사진

### 5.1.2. iGAN(Interactive Image Generation via GAN)

이미지 생성 인터페이스인 iGAN(Interactive Image Generation via GAN)은 간단한 스케치로 이미지를 자동으로 생성해준다.

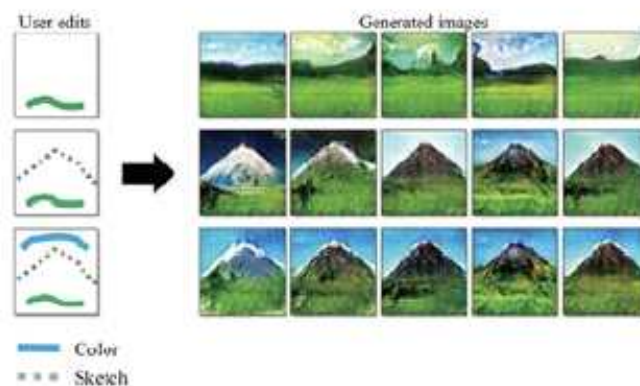


그림 20 | iGAN

### 5.1.3. GAN을 활용한 이미지 복원 : SRGAN

화질이 선명하지 않은 사진은 GAN을 통해서 복원하는 기능을 제공할 수 있으며, SRGAN을 통해 실현할 수 있다.



그림 21 | SRGAN

그뿐만 아니라 간단한 스케치만으로 제품의 디자인 시안을 생성하고, 내가 그린 그림을 유명 화가의 화풍으로 생성해주는 등의 분야에 활용할 수 있다.

### 5.2. Eye In-Painting

2017년 12월 페이스북(Facebook)에서 ExGAN이라는 기술을 개발하였다. 이는 사진을 찍었을 때, 눈을 감은 경우 눈을 뜬 사람의 모습으로 사진을 바꿔주는 과정을 공개하였다. RealEye Opener라는 이름으로 불리는 이 기술은 GAN을 이용해 진짜 같은 가짜를 만들어 눈을 감은 사진에 합성을 시켜준 것이다. 특정 장소나 다시 찍을 수 없는 경우인 사진에서 눈을 감은 경우, 이 사진을 보정할 때에 사용할 수 있다.



그림 22 | 눈을 감을 사진을 뜬 사진으로 합성한 예제

### 5.3. 영상 합성

2017년 8월 미국의 워싱턴 대학교의 연구진은 버락 오바마 전 대통령의 가짜 영상을 제작하여 공개하였다. 이는 오바마 전 대통령의 과거 연설 영상들에서 음성을 따고, 이 음성에 맞는 입 모양을 만들어 제작되었다. 논문의 저자는 먼저, 오디오 Input

을 시간에 따라 달라지는 입 모양으로 변환시킨 후, 진짜 같은 입 모양을 생성해 이를 대상 비디오의 입 부분에 삽입하는 방식으로 생성하였다고 한다. 이를 통해 GAN은 음성 합성에도 효과를 내고 있음을 확인할 수 있다.



Output Obama Video

그림 23 | 미국 전 대통령 버락 오바마의 가짜 영상

#### 5.4. 텍스트 생성

MIT의 연구진은 이미지와 시를 쌍으로 학습시켜 AI가 이미지를 보고, 이미지에 적절한 시를 만들어내는 연구를 진행하였다. 또한 GAN에 특정 문장의 스타일이나, 화법 등을 학습시켜 시를 제작하는 경우, 사람보다 더욱 풍부하고 정교한 언어를 사용하여 시를 만들어낼 수 있다.

#### 5.5. 인공지능 개발

2017년 7월 구글의 딥마인드는 사람과 물체의 보행 능력을 흉내 내는 인공지능을 개발하였다. 이 기술은 과거 알파고의 자체대국에서 활용된 강화학습을 적용하였으며, 보상함수를 단순한 승/패 여부가 아니라 행위의 적절성으로 표현한다는 점에 차이점을 두고 있다.

#### 5.6. 기타 및 국내 서비스

##### 5.6.1. Adobe

Adobe에서는 인물의 나이 및 표정과 같은 감정 상태를 조절할 수 있는 필터를 개발하여 자사 애플리케이션인 포토샵(Photoshop v.22)에 기능을 추가하였다.

##### 5.6.2. Google

구글은 공상과학(SF) 영화 속의 가상 생명체를 실감이 나게 제작할 수 있는 도

구인 Chimera Painter을 개발하였다. 이는 자체 GAN 모델인 CreatureGAN을 이용해 생물의 구조적인 특징을 학습하고, 새로운 유형의 생물체를 사실적으로 표현하는 것이 가능하도록 하는 기술이다.

### 5.6.3. 딥페이크 영상을 감지하는 프로그램

미국 스탠퍼드 대학교 HAI(Human-centered AI) 연구팀은 GAN을 이용해 조작된 영상 속 인물의 입 모양과 음성 소리 간의 미세한 불일치를 81%, 최대 96% 수준의 정확도로 파악하는 시스템을 개발하였다. 영상과 음성의 10초 단위 무작위 동기화 검사를 병행하며 정확도를 높였으며 이는 AI로 AI의 악용사례를 막는 대표적인 예시 중 하나이다.

### 5.6.1. NAVER의 몰입형 웹툰 ‘마주쳤다’

네이버 웹툰 ‘마주쳤다’는 GAN을 적용한 몰입형 웹툰을 공개했다. 이는 독자가 사진을 찍어 올려주면 자신의 얼굴이 웹툰 화 되어 만화에 등장하도록 하는 서비스를 통해 마치 독자가 웹툰 속 주인공이 되는 경험을 제공하였다.

### 5.6.2. NAVER의 스타트업 D2 스타트업 팩토리(D2SF)

네이버의 D2 스타트업 팩터리의 알레시오는 딥러닝 GAN기술을 기반으로 태아의 입체 초음파 사진을 기반으로 생후 예상 사진으로 변환해주는 솔루션을 개발하고 있다.

## 6. GAN 악용사례

GAN은 실제에 가까운 가짜 데이터를 생성하는 데에 목표를 두고 있다. 따라서 실제와 구별되지 않기 때문에 거짓이 현실을 압도하여 해당 기술을 악용하는 사례가 증가할 수 있다.

### 6.1. 딥 페이크

딥 페이크란, 인공지능(AI) 기술 중 하나인 딥러닝의 딥(Deep)과 가짜(Fake)의 합성어로 사진과 영상을 인공지능 기술로 자동 변조하는 CG 기술 중 하나이며, 생성적 적대 신경망인 GAN이 활용된다.

현재 GAN을 활용해 ‘딥 페이크’ 포르노 영상을 제작하고, 유통하는 부작용이 늘어나고 있다. 유명 연예인의 얼굴을 포르노 영상에 합성하는 것이 GAN이라는 기술을 바탕으로 전보다 정교해지면서 디지털 성범죄의 문제가 무분별하게 커지고 있다.

## 6.2. 가짜 뉴스

과거에도 가짜 뉴스에 대한 문제는 생활 속에 함께했으나, 텍스트보다 더욱 신빙성이 있으며 쉽고, 빠르게 전할 수 있는 이미지나 음성, 영상들이 GAN을 이용해 쉽게 제작된다면 가짜 뉴스가 퍼지는 문제를 더욱 심각하게 만들 수 있다. IT 자문기관인 가트너는 2018년 이후 10대 디지털 기술 전망 발표에서 2022년부터 사람들은 실제 정보보다 AI가 만든 허위 정보를 더욱 많이 접할 것으로 전망했다.

# V. TF2-GAN

## 1. TF2-GAN

TF2-GAN은 tensorflow2에서 실행되는 GAN으로 노이즈(Noise) 속에서 숫자를 도출해낸다. 해당 프로그램을 실행시키지 위해 MNIST(Mixed National Institute of Standards and Technology) 숫자 손글씨 데이터 셋을 사용하였다.

### 1.1. MNIST Dataset

- ① LeCun 교수가 만든 데이터 셋
- ② 60,000개의 트레이닝 셋과 10,000개의 테스트 셋으로 구성
- ③ 트레이닝 셋 : 학습 데이터로 사용
- ④ 테스트 셋 : 신경망을 검증하는 데에 사용
- ⑤ 숫자는 0에서 1까지의 값을 가지며, 크기는 고정
- ⑥ 크기는 28\*28px로 표준화되어있으며 중심에 배치



그림 24 | MNIST 데이터셋

## 2. TF2-GAN 실행

### 2.1. 실행 순서

- ① git clone 명령어를 이용해 오픈 소스를 불러옴
- ② venv 가상환경 구축
- ③ tensorflow 2.0 버전과 tensorflow dataset, matplotlib 패키지 다운로드
- ④ TF2-GAN 중 gan 폴더의 train.py 파일 실행

### 2.2. 코드 분석

#### 2.2.1. 분석 1

latent_dim(latent space demension)	GAN generation을 위한 잠재 공간의 차원
epoch	데이터 셋 iteration 횟수
batch_size	gradient를 구현하기 위해 한 번에 학습할 크기
buffer_size	데이터 셋 셔플을 위해 데이터를 임시로 담기 위한 공간의 사이즈
save_interval	중간에 학습이 예기치 않게 중단되더라도 학습한 내용을 유지할 수 있도록 함

```
# settting hyperparameter
latent_dim = 100
epochs = 800
batch_size = 200
buffer_size = 6000
save_interval = 50
```

그림 25 | 코드 1

### 2.2.2. 분석 2

① optimizer.Adam() 함수 사용

: 이 함수는 경사 하강법의 개선된 버전이다.

② 학습률(learning rate) = 0.0002

: 학습률은 특정 가중치 값을 줬을 때, 기울기에 학습률을 곱한 다음 가중치를 결정하기 때문에 너무 크지도, 작지도 않은 적당한 값을 가져야 한다. 만약, 학습률이 너무 커졌을 때 가중치의 값이 발산하게 되며 이러한 현상을 Overshooting 이라고 한다. 반대로 학습률이 작아졌을 때 반복의 횟수가 증가한다. 따라서 학습률(learning rate)의 최적값을 찾는 것은 중요한 작업이다.

③ 손실함수는 cross entropy loss 사용

```
gen_optimizer = tf.keras.optimizers.Adam(0.0002)
disc_optimizer = tf.keras.optimizers.Adam(0.0002)
```

```
train_dataset = train_data.map(normalize).shuffle(buffer_size).batch(batch_size)
cross_entropy = tf.keras.losses.BinaryCrossentropy(from_logits=True)
```

그림 26 | 코드 2

### 2.3. 실행 결과

실행 결과는 다음과 같이 나온다. 왼쪽의 그림은 epoch가 0인 이미지이고, 오른쪽은 epoch가 750인 이미지이다. MNIST 데이터 셋을 이용하여 노이즈(Noise)로부터 손글씨를 도출해 낸 것을 알 수 있다.

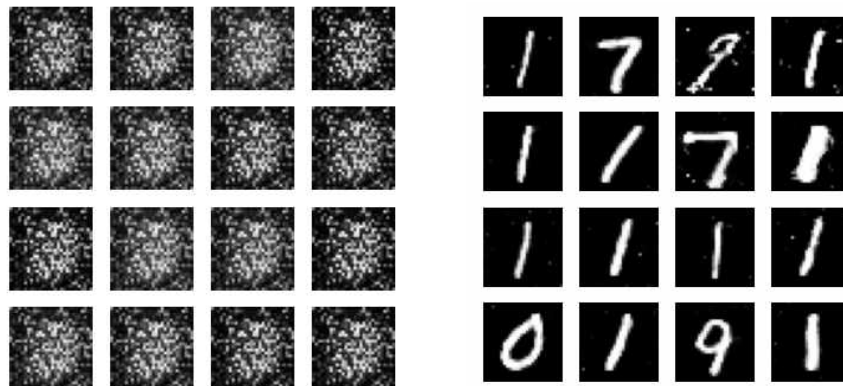


그림 27 | 왼쪽\_epoch 0, 오른쪽\_epoch 750



## VI. Toonify Yourself(Toonification GAN)

### 1. Toonify Yourself

Toonify Yourself는 StyleGAN2를 사용한 것으로, 사람의 얼굴을 디즈니 캐릭터의 얼굴처럼 변환시켜주는 GAN이다. 이는 StyleGAN 인코더를 사용해 실제 얼굴에서 실제 얼굴이 가지고 있는 잠재된 표현을 찾아 혼합된 모델을 사용해 표현하기에 놀라운 준실제 만화화 결과를 얻어낼 수 있다.

#### 1.1. 코드 준비 단계

디즈니 얼굴로 변환된 이미지를 얻기 위해 제공된 도커(Docker) 파일을 사용하였으며, preload.sh 파일과 toonify-yourself.py 파일을 만들어 코드를 실행하였다.

preload.sh 파일은 미리 설치되어 있어야 하는 라이브러리나, 불러올 파일들을 입력하는 명령어를 정리하였으며, toonify-yourself.py 파일은 실제로 사람의 사진을 디즈니 얼굴처럼 만들어줄 때 필요한 명령어를 정리하였다. url 다운로드 문제를 해결하기 위해 ffhq-cartoon-blended-64.pkl 파일과 stylegan2-ffhq-config-f.pkl 파일을 미리 다운로드 받아야 했으며 해당 파일은 아래의 url을 통해 다운받을 수 있다. toonify-yourself.py 파일의 blended-url의 ffhq-cartoon-blended-64.pkl 파일의 경로를 절대경로로 변경해주어야 하며, ffhq-url에는 stylegan2-ffhq-config.pkl 파일의 절대경로를 써넣어야 한다.

-[https://drive.google.com/uc?id=1H73TfV5gQ9ot7slSed\\_l-lim9X7pMRiU](https://drive.google.com/uc?id=1H73TfV5gQ9ot7slSed_l-lim9X7pMRiU)

-<http://d36zk2xti64re0.cloudfront.net/stylegan2/networks/stylegan2-ffhq-config-f.pkl>

##### 1.1.1. preload.sh

도커(docker) 컨테이너 안에서 작업한 것으로 tensorflow와 numpy, cuda, nudnn 패키지는 미리 설치가 되어있는 환경에서 진행하였다.

```
#pip install tensorflow_version==1.15
#pip install tensorflow-gpu==1.15 numpy
git clone https://github.com/justinpinkney/stylegan2
cd stylegan2
nvcc test_nvcc.cu -o test_nvcc -run

mkdir raw
mkdir aligned
mkdir generated

wget https://www.bntnews.co.kr/data/bnt/image/201008/cba489d1b74edb358cfcc9fe5c84d23c.jpg -O raw/example.jpg

python align_images.py raw aligned

python project_images.py --num-steps 500 aligned generated
```

그림 28 | preload.sh 명령어



그림 28의 마지막 2줄의 명령어는 toonify-yourself.py 파일을 실행한 후에 입력해야 한다. import로 시작하는 명령어는 toonify-yourself.py에 있는 명령어이며, python으로 시작하는 두 개의 명령어는 preload.sh에 있는 명령어이다.

toonify-yourself.py를 실행하기 위해서는 preload.sh에 있는 파일들과 라이브러리들은 미리 존재해야 하며, python으로 시작하는 두 명령어는 toonify-yourself.py에 있는 명령어이기 때문에 해당 명령어를 실행하기 위해서는 preload.sh 파일을 한 번씩 더 실행을 시켜주어야 한다.

즉, GAN을 정상적으로 동작시키기 위해서는 preload.sh 파일의 wget 명령어 부분까지 실행을 시킨 후, toonify-yourself.py 파일을 실행시켜야 한다. 이후, python으로 시작하는 두 개의 명령어를 실행시킨 후, 다시 toonify-yourself.py 파일을 실행시키는 순서로 진행되어야 한다.

```
[ ] import pretrained_networks

# use my copy of the blended model to save Doron's download bandwidth
# get the original here https://mega.nz/folder/Ot1lJwa#C947mCCdEfMCRtWnDcs4qW
blended_url = "https://drive.google.com/uc?id=1H73TFv5g09ot7sISed_l-lin8X7eMRiU"
ffhq_url = "http://d36zk2xti84re0.cloudfront.net/stylegan2/networks/stylegan2-ffhq-config-f.pkl"

_, Gs_blended = pretrained_networks.load_networks(blended_url)
_, Gs = pretrained_networks.load_networks(ffhq_url)

[ ] !python align_images.py raw aligned

[ ] !python project_images.py --num-steps 500 aligned generated

[ ] import numpy as np
from PIL import Image
import dnnlib
import dnnlib.tflib as tflib
from pathlib import Path

latent_dir = Path("generated")
latents = latent_dir.glob("*.npy")
for latent_file in latents:
    latent = np.load(latent_file)
    latent = np.expand_dims(latent, axis=0)
    synthesis_kwargs = dict(output_transform=dict(func=tflib.convert_images_to_uint8, nchw_to_nhwc=False), minibatch_size=8)
    images = Gs_blended.components.synthesis.run(latent, randomize_noise=False, **synthesis_kwargs)
    Image.fromarray(images.transpose((0,2,3,1))[0], 'RGB').save(latent_file.parent / (f"{latent_file.stem}-toon.jpg"))
```

그림 29 | 명령어의 올바른 순서

## 1.2. toonify-yourself.py

```
import sys
sys.path.append('/stylegan2')

import tensorflow as tf
gpus = tf.config.experimental.list_physical_devices('GPU')

if gpus:
    print(gpus)
    try:
        for gpu in gpus:
            tf.config.experimental.set_memory_growth(gpu, True)
    except RuntimeError as e:
        print(e)

import pretrained_networks

# use my copy of the blended model to save Doron's download bandwidth
# get the original here https://mega.nz/folder/Ot1IzJwa#C947mCCdEfMCRTWhDcs4qW
blended_url = "/stylegan2/ffhq-cartoon-blended-64.pkl"
ffhq_url = "/stylegan2/stylegan2-ffhq-config-f.pkl"

_, _, Gs_blended = pretrained_networks.load_networks(blended_url)
_, _, Gs = pretrained_networks.load_networks(ffhq_url)

import numpy as np
from PIL import Image
import dnnlib
import dnnlib.tflib as tflib
from pathlib import Path

latent_dir = Path("generated")
latents = latent_dir.glob("*.npy")
for latent_file in latents:
    latent = np.load(latent_file)
    latent = np.expand_dims(latent, axis=0)
    synthesis_kwargs = dict(output_transform=dict(func=tflib.convert_images_to_uint8, nchw_to_nhwc=False), minibatch_size=8)
    images = Gs_blended.components.synthesis.run(latent, randomize_noise=False, **synthesis_kwargs)
    Image.fromarray(images.transpose((0,2,3,1))[0], 'RGB').save(latent_file.parent / (f"{latent_file.stem}-toon.jpg"))

from IPython.display import Image, display
embedded = Image(filename="generated/example_01.png", width=256)
tooned = Image(filename="generated/example_01-toon.jpg", width=256)
```

그림 30 | toonify-yourself.py 명령어

### ① tf.config.experimental.set\_memory\_growth(gpu, True)

: GPU 메모리가 부족할 때 사용하는 명령어로, GPU 메모리를 조금씩 할당해줄 때 사용

- ② `_, _, Gs_blended = pretrained_networks.load_networks(blended_url)`  
`_, _, Gs = pretrained_networks.load_networks(ffhq-url)`  
 : 리스트로 된 `blended_url`을 3개의 변수에 넣어주는 명령어
- ③ `images = Gs_blended.components.synthesis.run(latent, randomize_noise`  
`= False, **synthesis_kwargs)`  
 : 디즈니 화 한 이미지를 `images` 변수에 저장하는 명령어

## 2. Trouble Shooting

구글의 Colab 환경과 RTX 3080 GPU 환경에서 Toonify yourself 실행 시 발생했던 오류, Trouble Shooting을 정리한다.

### 2.1. Colab 환경

구글의 Colab은 웹 브라우저에서 텍스트와 프로그램 코드를 자유롭게 작성하고, 실행할 수 있도록 도와주는 일종의 온라인 텍스트 에디터이며, 클라우드 기반의 주피터 노트북 개발 환경이다. 머신러닝을 공부하기 위해서는 컴퓨터의 사양이 굉장히 중요 한데, 구글 코랩(Colab)은 소지한 컴퓨터 대신 CPU와 RAM을 제공해주기에 컴퓨터의 성능과는 상관없이 프로그램 실행을 진행할 수 있도록 도와준다. 코랩(Colab)에서의 코랩 파일은 노트북이라고 부르며, 코드를 입력하는 곳은 코드의 셀이라고 부르고, 코랩 환경에서 사용할 수 있는 프로그래밍언어는 파이썬(python)이다.

#### 2.1.1. %tensorflow\_version 1.x 오류

해당 에러는 tensorflow 버전과 관련된 오류이다. toonify yourself 파일을 돌리기 위해서는 tensorflow의 버전이 1인 환경에서 동작하지만, Colab 환경은 tensorflow 2 버전을 제공하기에 해당 오류가 발생함을 확인하였다. 해당 버전을 1로 변경함으로써 해당 오류를 해결할 수 있었다.



```

%tensorflow_version 1.x

ValueError                                Traceback (most recent call last)
<ipython-input-18-8d2919c1d83c> in <module>
----> 1 get_ipython().run_line_magic('tensorflow_version', '1.x')

1 frames
/usr/local/lib/python3.7/dist-packages/google/colab/tensorflow_magics.py in _tensorflow_version(line)
   39
   40     Your notebook should be updated to use Tensorflow 2.
--> 41     See the guide at https://www.tensorflow.org/guide/migrate#migrate-from-tensorflow-1x-to-tensorflow-2.
   42
   43
ValueError: Tensorflow 1 is unsupported in Colab.

Your notebook should be updated to use Tensorflow 2.
See the guide at https://www.tensorflow.org/guide/migrate#migrate-from-tensorflow-1x-to-tensorflow-2.

```

그림 31 | 오류 발생 화면

```
✓ [2] !pip uninstall tensorflow -y
1분 !pip install tensorflow-gpu==1.15
!apt install --allow-change-held-packages libcudnn7=7.4.1.5-1+cuda10.0
```

그림 32 | 버전 변경 코드

### 2.1.2. invalid load key, '<'. 오류

구글 드라이브에서 url을 가져오는 부분에서 invalid load key, '<'. 의 오류가 발생하였다. 이는 구글 드라이브에 있는 파일의 크기가 2기가가 넘어갔을 때, 바로 다운로드가 이루어지지 않기 때문에 발생하는 오류이다. 따라서 해당 문제를 해결하기 위해 구글 드라이브의 url 정보를 제공하는 것이 아닌, 파일을 다운로드받아 절대경로를 제공하는 방법으로 해결하였다.

그러나 절대경로로 변경한 후에도 파일이 올라가지 않는 등의 파일과 관련한 오류가 계속 발생하였다. 또한 Colab 환경에서는 이미 라이브러리가 자동으로 설치되어 있기에 버전을 맞추는 데에 까다로움이 존재하여 RTX 3080 GPU에서 실행시키는 방법을 차선택으로 선택하였다.

```
0분 import pretrained_networks
import pickle5 as pickle

blended_url = '/content/ffhq-cartoon-blended-64.pkl'
ffhq_url = "/content/stylegan2-ffhq-config-f.pkl"

_, _, Gs_blended = pretrained_networks.load_networks(blended_url)
_, _, Gs = pretrained_networks.load_networks(ffhq_url)

UnpicklingError                                Traceback (most recent call last)
<ipython-input-61-c9713fcea3d> in <module>
      5 ffhq_url = "/content/stylegan2-ffhq-config-f.pkl"
      6
----> 7 _, _, Gs_blended = pretrained_networks.load_networks(blended_url)
      8 _, _, Gs = pretrained_networks.load_networks(ffhq_url)

/content/stylegan2/stylegan2/pretrained_networks.py in load_networks(path_or_gdrive_path)
     74     tflib.init_tf()
     75     with stream:
--> 76         G, D, Gs = pickle.load(stream, encoding='latin1')
     77     _cached_networks[path_or_url] = G, D, Gs
     78     return G, D, Gs

UnpicklingError: pickle data was truncated
```

그림 33 | 오류 발생 화면

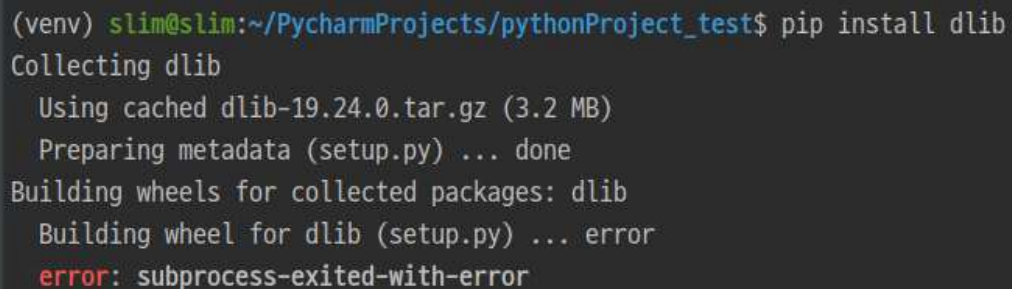
## 2.2. RTX 3080 GPU 환경 (Pycharm 이용)

Toonify yourself의 버전을 더욱 쉽게 맞추기 위해 RTX 3080 GPU 환경에서 실행하였다. 파이썬 3.10.6 환경에서 실행하였으며, 가상환경(venv)을 만들어 필요한 패키지들을 모두 설치해 환경을 구축하였다.

### 2.2.1. dlib 오류

dlib 오류를 해결하기 위해 venv 환경에서 오류를 해결하려고 해보았으나, 이는 anaconda 가상환경을 설정하여 해당 환경에서 실행해야 하였다.

- ① anconda 가상환경 설정
- ② -m pip install --upgrade pip : pip 업그레이드
- ③ pip install cmake : cmake 패키지 설치
- ④ pip install dlib : dlib 패키지 설치

A terminal window showing the command 'pip install dlib' being executed in a virtual environment. The output shows that dlib-19.24.0.tar.gz (3.2 MB) was collected and metadata was prepared. However, when building the wheel for dlib, an error occurred: 'error: subprocess-exited-with-error'.

```
(venv) slim@slim:~/PycharmProjects/pythonProject_test$ pip install dlib
Collecting dlib
  Using cached dlib-19.24.0.tar.gz (3.2 MB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: dlib
  Building wheel for dlib (setup.py) ... error
error: subprocess-exited-with-error
```

그림 34 | 오류 발생 화면

### 2.2.2. python align\_images.py raw aligned 오류

python align\_images.py raw aligned 부분에서 오류가 발생하였으며, 이는 tensorflow와 pillow 패키지의 재설치를 통해 해결하였다.

### 2.2.3. tensorflow 버전 오류

tensorflow 버전과 관련된 오류가 반복해서 발견되는 것을 확인한 후, 원인에 대해 검색해 본 결과 파이썬(python) 과 tensorflow 버전의 문제라는 것을 확인할 수 있었다.

아래 그림 35를 보면 tensorflow==1.15 버전과 호환을 할 수 있는 Python 버전을 확인할 수 있다. 따라서 파이썬 3.10.6 버전이 아닌 python 3.7 버전으로 변경하여 진행하였다.

### Programming Language

- [Python :: 2](#)
- [Python :: 2.7](#)
- [Python :: 3](#)
- [Python :: 3.4](#)
- [Python :: 3.5](#)
- [Python :: 3.6](#)
- [Python :: 3.7](#)

그림 35 | tensorflow==1.15 호환 가능 python 버전

#### 2.3.4. raise RuntimeError('No GPU devices found') 오류

tensorflow와 Python 버전을 맞춘 후, 파일을 실행시켰으나 CUDA 버전과 관련된 오류가 발생하였다.

사용하고 있는 GPU에서는 CUDA 11 버전과 tensorflow==1.15 버전을 사용하고 있었으나 그림 36을 보면 사용하고 있는 tensorflow==1.15버전에서는 CUDA 11을 버전을 사용할 수 없었고, Python 또한 3.6 버전을 사용하는 것이 좋을 수 있다. 현재 동작하고 있는 서버의 버전 변경은 다른 파일들에도 영향을 끼칠 수 있으며, 해당 GPU에서는 CUDA 11 버전을 지원하지 않기에 도커(docker)를 사용해 새로운 환경을 구축하여 실행하였다.

```
raise RuntimeError('No GPU devices found')
RuntimeError: No GPU devices found
```

그림 36 | 오류 발생 화면

I quote this: [Release Notes :: CUDA Toolkit Documentation](#) <sup>895</sup> [CUDA Toolkit v11.1.0][Release Notes]

"Added support for NVIDIA Ampere GPU architecture based GA10x GPUs (compute capability 8.6), including the GeForce RTX-30 series."

The answer I believe is no, because support for SM\_86 was added in CUDA 11.x. This link might help you understand better: [Matching CUDA arch and CUDA gencode for various NVIDIA architectures - Arnon Shimoni](#) <sup>585</sup>

If you could point out why you need CUDA 10.1 specifically I might be able to help you.

그림 37 | 오류 확인 화면 (RTX 3080 GPU CUDA 10 버전 지원 불가)



## 2.3. RTX 2070 SUPER GPU 환경

CUDA 10 버전을 활용하기 위해 도커(docker)를 사용하였다. 도커(docker)란, 리눅스의 응용 프로그램들을 프로세스 격리 기술들을 사용하여 컨테이너로 실행하고, 관리하는 오픈소스 프로젝트를 말한다. 보통 이미지를 pull 한 후, 이미지를 토대로 컨테이너를 만들거나 DockerFile을 이용해 build를 하여 이미지를 생성하기도 한다.

StyleGAN2에 대한 내용이 적힌 깃허브(NVlabs/stylegan2)에 버전 환경이 올바르게 구축된 도커 파일을 사용하여 pull하여 이미지 컨테이너를 만들어주어 실행하였다.

dnlib	Workaround for NCCL bug in TF 1.15	2 years ago
docs	Update versions.html for StyleGAN3	13 months ago
metrics	Update S3 links.	2 years ago
training	Initial commit	3 years ago
.gitignore	Convert seed 'range' into a list	3 years ago
<u>Dockerfile</u>	Workaround for NCCL bug in TF 1.15	2 years ago
LICENSE.txt	Initial commit	3 years ago
README.md	Update README.md	13 months ago
dataset_tool.py	Initial commit	3 years ago
pretrained_networks.py	Update S3 links.	2 years ago
projector.py	Update S3 links.	2 years ago

그림 38 | 사용 Docker 파일

### 2.3.1. git 설치 오류

도커 환경을 만든 후, git clone 명령어를 사용해 GitHub repository에 있는 내용을 가져오기 위해서 'apt-get install git'을 통해 git을 설치하려 했으나 오류가 발생하였다. 이는 'apt-get update'와 'apt-get -fix-missing'을 통해 해결하여 git을 설치하였다.

### 2.3.2. NameError: name 'display' is not defined 오류

NameError: name 'display' 오류는 파일로 저장되지 않았을 경우 한 번 더 확인할 수 있도록 화면에 보여주는 'display(embedded)'와 'display(tooned)' 코드를 주석처리를 함으로써 해결하였다.

```

root@2c40148b0c05:/stylegan2# python toonify_yourself.py
Setting up TensorFlow plugin "fused_bias_act.cu": Preprocessing... Loading... Done.
Setting up TensorFlow plugin "upfirdn_2d.cu": Preprocessing... Loading... Done.
2022-11-19 05:43:24.480586: E tensorflow/stream_executor/cuda/cuda_driver.cc:828] failed to allocate 3.126 (3353522
944 bytes) from device: CUDA_ERROR_OUT_OF_MEMORY: out of memory
Traceback (most recent call last):
  File "toonify_yourself.py", line 34, in <module>
    display(embedded)
NameError: name 'display' is not defined

```

그림 39 | 오류 발생 화면

```

from IPython.display import Image
embedded = Image(filename="generated/example_01.png", width=256)
#display(embedded)
tooned = Image(filename="generated/example_01-toon.jpg", width=256)
#display(tooned)

```

그림 40 | 오류 해결 화면 (주석처리)

### 2.3.3. CUDA\_ERROR\_OUT\_OF\_MEMORY: out of memory 오류

해당 오류는 아래의 그림 41의 코드를 추가하며 해결할 수 있었다.

```

import tensorflow as tf
gpus = tf.config.experimental.list_physical_devices('GPU')

if gpus:
    print(gpus)
    try:
        for gpu in gpus:
            tf.config.experimental.set_memory_growth(gpu, True)
    except RuntimeError as e:
        print(e)

```

그림 41 | 해결 코드



### 3. 결과

#### 3.1. 최종 라이브러리 버전

- absl-py==0.7.1
- asn1crypto==0.24.0
- astor==0.8.0
- backcall==0.2.0
- certifi==2022.9.24.
- chardet==3.0.4
- cmake==3.24.3
- cryptography==2.1.4
- decorator==5.1.1
- dlib==19.24.0
- gast==0.2.2
- google-pasta==0.1.7
- grpcio==1.21.1
- h5py==2.9.0
- idna==2.6
- ipython==7.16.3
- ipython-genutils==0.2.0
- jedi==0.17.2
- Keras-Applications==1.0.8
- Keras-Preprocessing==1.1.0
- Keyrings.alt==3.0
- Markdown==3.1.1
- numpy==1.16.4
- parso==0.7.1
- pexpect==4.8.0
- pickleshare==0.7.5
- Pillow==6.2.1
- prompt-toolkit==3.0.32
- protobuf==3.8.0
- ptyprocess==0.7.0
- pycrypto==2.6.1
- Pygments==2.13.0
- pygobject==3.26.1
- python-apt==1.6.5+ubuntu0.7

- pyxdg==0.25
- requests==2.22.0
- scipy==1.3.3
- SecretStorage==2.3.1
- six==1.11.0
- tensorboard==1.14.0
- tensorflow-estimator==1.14.0
- tensorflow-gpu==1.14.0
- termcolor==1.1.0
- traitlets==4.3.3
- urllib3==1.25.11
- wcwidth==0.2.5
- werkzeug==0.15.4
- wrapt==1.11.2

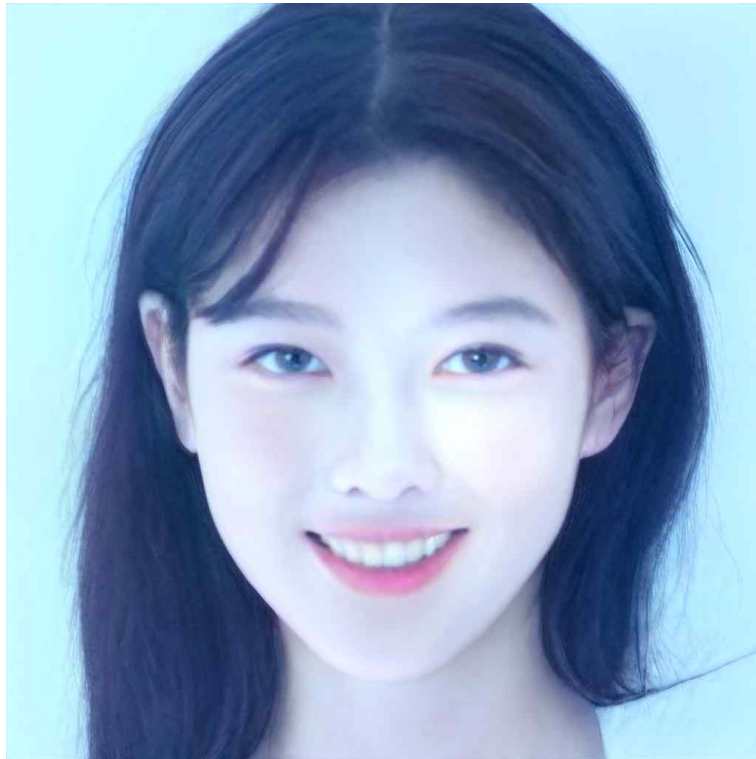
## 3.2. 출력 결과

### 3.2.1. 배우 김유정

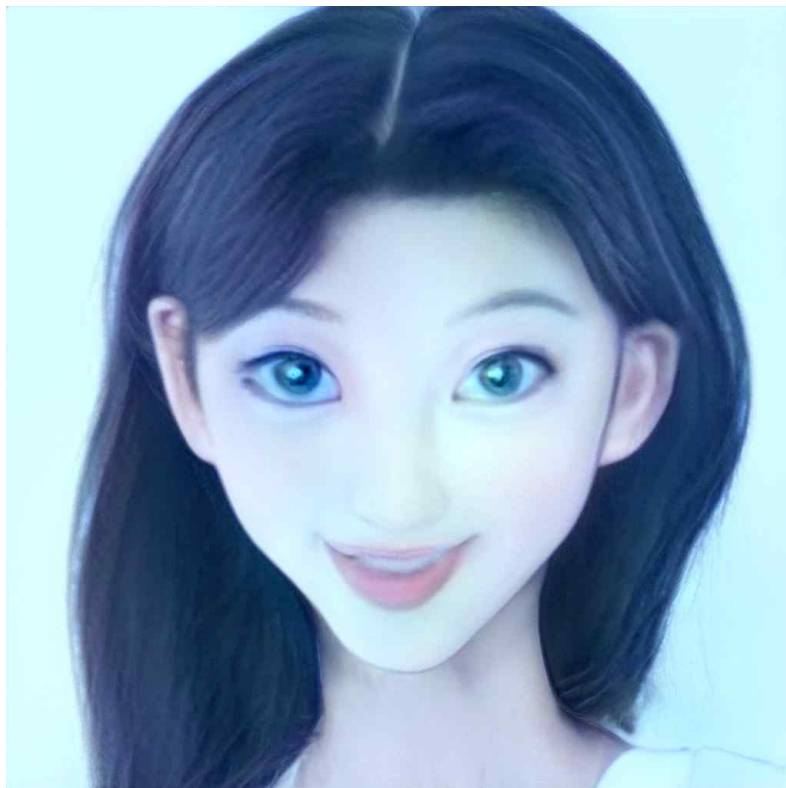
#### ① 배우 김유정



② blended 사진(ffhq-cartoon-blended-64.pkl 적용)



③ 결과 사진(stylegan2-ffhq-config-f.pkl 적용)



### 3.2.2. 코미디언 박명수

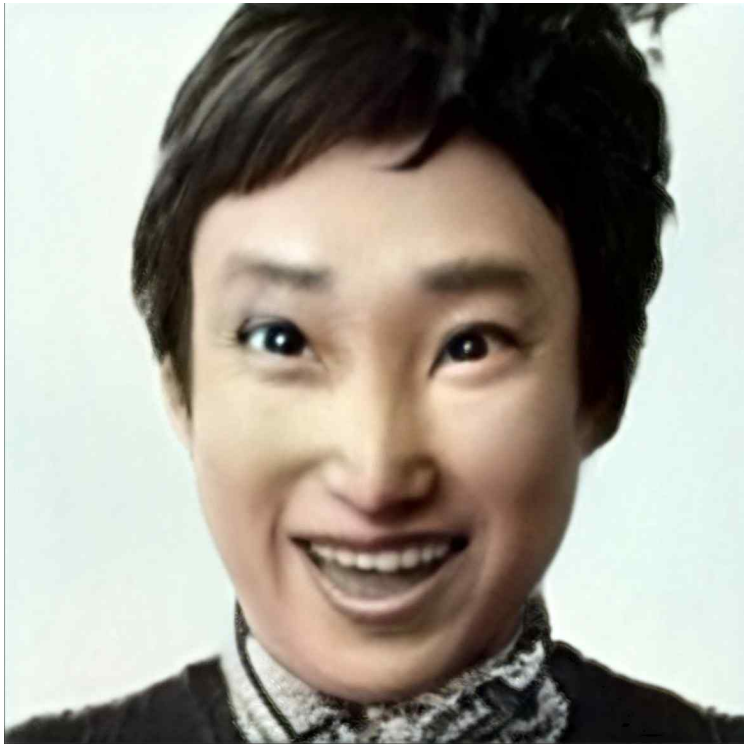
#### ① 원본 사진



#### ② blended 사진(ffhq-cartoon-blended-64.pkl 적용)



③ 결과 사진(stylegan2-ffhq-config-f.pkl 적용)



## VII. 결론

### 1. 결론

- 최근 인공지능과 관련된 수요가 점차 증가하고 있다. 그중에서도 기존에 정답을 줘야 하는 지도 학습과는 다르게 정답을 미리 제공하지 않아도 학습이 가능한 비지도 학습에 관한 관심과 연구가 증가하고 있다.
- 비지도 학습의 한 종류의 GAN(Generative Adversarial Networks)에 관한 연구를 진행하며, 이는 인공지능의 목표인 미래 예측과 연구 분야로의 가능성이 충분함을 인지할 수 있었다.
- 업계에서 자사의 제품이나 서비스에 인공지능의 적용과 활용이 요구되고 있다. 음성인식, 헬스케어, 자율주행 자동차, 이미지 인식 등의 일상생활 속에 인공지능의 지분은 많이 녹아있음을 알 수 있다.
- 심층 신경망이라고 불리는 딥러닝(Deep Learning)은 수동적인 특징 추출을 요구하지 않으며, 직접 기능을 학습하는 신경망 구조로 되어있다.
- 신경망의 종류는 ANN(Artificial Neural Network), DNN(Deep Neural Network), CNN(Convolutional Neural Networks), RNN(Recurrent Neural Network), LSTM(Long Short Term Memory) 등이 있다.

- GAN은 진짜 같은 가짜 데이터를 만들어 내는 인공지능(AI) 기술이다. 이는 생성(Generator) 신경망과 판별(Discriminator) 신경망이 서로 적대적으로 학습되며 더욱 정교한 거짓 데이터는 만들게 된다.
- 기존 전문가가 진행하던 포토샵과 같은 기술적인 부분을 GAN을 활용하면 빠르고 쉽게 데이터를 만들어 낼 수 있다.
- 이미지뿐만 아니라 영상 합성이나 텍스트 생성과 같은 부분에서도 GAN을 활용할 수 있다.
- GAN을 이용한 악용사례 또한 증가하고 있다. 이는 딥페이크와 가짜 뉴스 등이 있으며 이를 막을 수 있는 제도나 서비스를 제공하는 것이 필요하다.
- TF2-GAN은 노이즈(Noise)에서 MNIST 데이터 셋을 활용해 글씨를 도출해내는 프로그램이다.
- TF2-GAN을 사용해 저화질 사진의 해상도를 높일 수 있으며, 나아가 훼손된 데이터를 복원하는 기능을 제공할 수 있다.
- Toonify Yourself는 디즈니 데이터 셋을 사용하여 사용자의 사진을 디즈니 캐릭터로 변환해주는 기능을 제공한다. 이를 활용해 나만의 캐릭터를 만들거나, 독창적인 개별적 특징을 살린 아바타를 제작할 수 있다.

## 2. 향후 연구 방향

- Toonify Yourself를 활용해 자신과 비슷한 캐릭터를 제작할 수 있으며, 이를 메타버스 상에서 구현할 수 있는 방향으로 향후 연구를 진행할 예정이다.
- 사용자의 얼굴뿐만 아니라 몸도 반영하여 적용하는 연구 또한 진행한다.
- 현재 2D로 만들어진 유니티 환경을 이용하여, 이를 3D로 구현하고자 한다.

## 참고 문헌

- [1] “딥러닝, 머신러닝과의 확연한 차이점은?,” 코딩월드뉴스, 17-Jan-2021. [Online]. Available:
- [2] I. Goodfellow et al., “Generative Adversarial Nets,” in Advances in Neural Information Processing Systems, 2014, vol. 27. [Online]. Available: <https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf>
- [3] “인공지능? 머신러닝? 딥러닝?,” Make precious life, 24-Feb-2021. [Online]. Available: <https://syj9700.tistory.com/35>. [Accessed: 23-Nov-2022].
- [4] “What is Artificial Intelligence (AI)?,” IBM. [Online]. Available: <https://www.ibm.com/cloud/learn/what-is-artificial-intelligence>. [Accessed: 23-Nov-2022].
- [5] “인공지능(AI)은 어떻게 발달해왔는가, 인공지능의 역사,” NVIDIA Blog Korea, 01-Feb-2017. [Online]. Available: [https://blogs.nvidia.co.kr/2016/03/13/history\\_of\\_ai/](https://blogs.nvidia.co.kr/2016/03/13/history_of_ai/). [Accessed: 23-Nov-2022].
- [6] “Superintelligence,” Wikipedia, 10-Oct-2022. [Online]. Available: <https://en.wikipedia.org/wiki/Superintelligence>. [Accessed: 23-Nov-2022].
- [7] “인공지능,” Wikipedia, 22-Nov-2022. [Online]. Available: <https://ko.wikipedia.org/wiki/%EC%9D%B8%EA%B3%B5%EC%A7%80%EB%8A%A5>. [Accessed: 23-Nov-2022].
- [8] “지도 학습,” Wikipedia, 07-Feb-2022. [Online]. Available: [https://ko.wikipedia.org/wiki/%EC%A7%80%EB%8F%84\\_%ED%95%99%EC%8A%B5](https://ko.wikipedia.org/wiki/%EC%A7%80%EB%8F%84_%ED%95%99%EC%8A%B5). [Accessed: 23-Nov-2022].
- [9] “강화 학습,” Wikipedia, 20-Jun-2022. [Online]. Available:
- [10] Artificial Intelligence: What's the difference between Deep Learning and reinforcement learning? (2021) Forbes. Forbes Magazine. Available at: <https://www.forbes.com/sites/bernardmarr/2018/10/22/artificial-intelligence-whats-the-difference-between-deep-learning-and-reinforcement-learning/?sh=dfb2347271e1> (Accessed: November 23, 2022).
- [11] 딥페이크(deep fake) 정의, GAN기술, 장점, 단점, 사례 (2021) 모두의향연. TISTORY. Available at: <https://feastforall.tistory.com/71> (Accessed: November 23, 2022).
- [12] <https://www.codingworldnews.com/news/articleView.html?idxno=2036>. [Accessed: 23-Nov-2022].
- [13] StyleGAN network blending (no date) Justin Pinkney. Available at:

- <https://www.justinpinkney.com/stylegan-network-blending/> (Accessed: November 23, 2022).
- [14] 구글 코랩(google colab)으로 머신러닝 프로그램 돌리기 : Github 예제 有 (2021) 혼자 공부하는 책. Available at: <https://hongong.hanbit.co.kr/%EA%B5%AC%EA%B8%80-%EC%BD%94%EB%9E%A9-google-colab%EC%9C%BC%EB%A1%9C-%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D-%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%A8-%EB%8F%8C%EB%A6%AC%EA%B8%B0/> (Accessed: November 23, 2022).
- [15] 적대적 생성신경망(generative adversarial network)의 소개와 활용 현황 - SPRI (no date) SPRI 소프트웨어정책연구소. Available at: [https://www.spri.kr/posts/view/21883?code=industry\\_trend](https://www.spri.kr/posts/view/21883?code=industry_trend) (Accessed: November 23, 2022).
- [16] [외부기고] [새로운 인공지능 기술 GAN] ① 스스로 학습하는 인공지능 (samsungsds.com)
- [17] 머신러닝 딥러닝, 아직도 헷갈리는 사람을 위한 완벽 개념 정리 (2021) 패스트캠퍼스 미디어. Available at: [https://media.fastcampus.co.kr/knowledge/data-science/ai\\_study/](https://media.fastcampus.co.kr/knowledge/data-science/ai_study/) (Accessed: November 23, 2022).
- [18] [IT열쇳말] gan(생성적 적대 신경망) (2018) (주)블로터앤미디어. Available at: <https://www.bloter.net/newsView/blt201806080001> (Accessed: November 23, 2022).
- [19] 머신러닝이란? (no date) 이란? - MATLAB & Simulink. Available at: <https://kr.mathworks.com/discovery/machine-learning.html> (Accessed: November 23, 2022).
- [20] (no date) First gan. Available at: <https://kangbk0120.github.io/articles/2017-07/first-gan> (Accessed: November 23, 2022).
- [21] Tech issue - 인공지능 기술의 개념과 최신 동향 \_AI 기술을 적용한 '3가지' 사례 (no date) 기술과혁신 웹진. Available at: <http://webzine.koita.or.kr/201802-technology/Tech-Issue-%EC%9D%B8%EA%B3%B5%EC%A7%80%EB%8A%A5-%EA%B8%B0%EC%88%A0%EC%9D%98-%EA%B0%9C%EB%85%90%EA%B3%BC-%EC%B5%9C%EC%8B%A0-%EB%8F%99%ED%96%A5-AI-%EA%B8%B0%EC%88%A0%EC%9D%84-%EC%A0%81%EC%9A%A9%ED%95%9C-%E2%80%98EA%B0%80%EC%A7%80%E2%80%99-%EC%82%AC%EB%A1%80> (Accessed: November 23, 2022).



- [21] [MBL]인공지능, 기계학습, 딥러닝 그리고 GPU 클라우드 (no date) Steemit. Available at: <https://steemit.com/mbl/@mbl/mbl-gpu> (Accessed: November 23, 2022).
- [22] What is Artificial Intelligence (AI)? (no date) IBM. Available at: <https://www.ibm.com/cloud/learn/what-is-artificial-intelligence> (Accessed: November 23, 2022).
- [23] 인공지능의 정의와 활용 방안 (no date) 인공지능의 정의와 활용 방안 | SAS KOREA. Available at: [https://www.sas.com/ko\\_kr/insights/analytics/what-is-artificial-intelligence.html#history](https://www.sas.com/ko_kr/insights/analytics/what-is-artificial-intelligence.html#history) (Accessed: November 23, 2022).
- [24] (현장)송창현 네이버랩스 대표 "D2SF, 창업생태계 조성에 유의미한 역할 기대" (2018) 플래텀. Available at: <https://platum.kr/archives/100889> (Accessed: November 23, 2022).

"Computing machinery and intelligence," Parsing the Turing Test, pp. 23-65, 2007.

"Web 3.0 시대를 위한 활성화 방안 및 도입 시 고려사항," Security & Intelligence 이글루코퍼레이션, 07-Jul-2022. [Online]. Available: <https://www.igloo.co.kr/security-information/web-3-0-%EC%8B%9C%EB%8C%80%EB%A5%BC-%EC%9C%84%ED%95%9C-%ED%99%9C%EC%84%B1%ED%99%94-%EB%B0%A9%EC%95%88-%EB%B0%8F-%EB%8F%84%EC%9E%85-%EC%8B%9C-%EA%B3%A0%EB%A0%A4%EC%82%AC%ED%95%AD/>. [Accessed: 23-Nov-2022].

[1] X. Chen, Y. Duan, R. Houthoof, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets," in Advances in Neural Information Processing Systems, 2016, vol. 29. [Online]. Available: <https://proceedings.neurips.cc/paper/2016/file/7c9d0b1f96aebd7b5eca8c3eda19ebb-Paper.pdf>

[1] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation." arXiv, 2017. doi: 10.48550/ARXIV.1710.10196.

"(현장)송창현 네이버랩스 대표 'D2SF, 창업생태계 조성에 유의미한 역할 기대,'"

플래텀, 24-May-2018. [Online]. Available: <https://platum.kr/archives/100889>. [Accessed: 23-Nov-2022].

“Gan(생성적 적대 신경망),” GAN(생성적 적대 신경망). [Online]. Available: <https://terms.naver.com/entry.naver?docId=5645660&cid=59088&categoryId=59096>. [Accessed: 23-Nov-2022].

“Generative adversarial nets (GAN) 1: Gan과 DCGAN 설명,” tyami's study blog, 16-Oct-2020. [Online]. Available: <https://tyami.github.io/deep%20learning/GAN-1-theory-GAN-DCGAN/>. [Accessed: 23-Nov-2022].

“Mocha's machine learning,” 딥러닝 GAN 튜토리얼 - 시작부터 최신 트렌드까지 GAN 논문 순서 | mocha's machine learning. [Online]. Available: <https://ysbsb.github.io/gan/2020/06/17/GAN-newbie-guide.html>. [Accessed: 23-Nov-2022].

“‘양날의 칼’ 알고리즘, 악용 막아 난제 해결 도구로 써야,” 중앙일보, 22-Oct-2021. [Online]. Available: <https://www.joongang.co.kr/article/25017395#home>. [Accessed: 23-Nov-2022].

“Introduction &nbsp;&nbsp;  machine learning &nbsp;&nbsp;  google developers,” Google. [Online]. Available: <https://developers.google.com/machine-learning/gan>. [Accessed: 23-Nov-2022].

“Introduction &nbsp;&nbsp;  machine learning &nbsp;&nbsp;  google developers,” Google. [Online]. Available: <https://developers.google.com/machine-learning/gan>. [Accessed: 23-Nov-2022].

“Overview of gan structure &nbsp;&nbsp;  machine learning &nbsp;&nbsp;  google developers,” Google. [Online]. Available: [https://developers.google.com/machine-learning/gan/gan\\_structure](https://developers.google.com/machine-learning/gan/gan_structure). [Accessed: 23-Nov-2022].

“[외부기고] [새로운 인공지능 기술 gan] ① 스스로 학습하는 인공지능,” [외부기고] [새로운 인공지능 기술 GAN] ① 스스로 학습하는 인공지능. [Online]. Available: <https://www.samsungsds.com/kr/insights/generative-adversarial-network-ai.html>. [Accessed: 23-Nov-2022].

“Gan 조사,” Google Docs. [Online]. Available: <https://docs.google.com/document/d/158SnFBekOi4Xf8kByX3vLLeZOeRJHFKsEJWIw5>

v5Ip0/edit?usp=sharing. [Accessed: 23-Nov-2022].

<https://doi.org/10.48550/arXiv.1703.10593>

“U-Net: Convolutional Networks for Biomedical Image Segmentation,” Lecture Notes in Computer Science, pp. 234-241, 2015.

[1]A. Odena, “Semi-Supervised Learning with Generative Adversarial Networks.” arXiv, 2016. doi: 10.48550/ARXIV.1606.01583.

“[딥러닝] 기울기 소실(vanishing gradient)의 의미와 해결방법,” Hey Tech, 25-Jul-2022. [Online]. Available: <https://heytech.tistory.com/388>. [Accessed: 23-Nov-2022].

[1]T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation.” arXiv, 2017. doi: 10.48550/ARXIV.1710.10196.

[1]P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks.” arXiv, 2016. doi: 10.48550/ARXIV.1611.07004.

[1]A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks.” arXiv, 2015. doi: 10.48550/ARXIV.1511.06434.

[1]T. Karras, S. Laine, and T. Aila, “A Style-Based Generator Architecture for Generative Adversarial Networks.” arXiv, 2018. doi: 10.48550/ARXIV.1812.04948.

Google colab. [Online]. Available: <https://research.google.com/colaboratory/faq.html?hl=ko>. [Accessed: 23-Nov-2022].

“위키백과:대문,” Wikipedia, 08-Nov-2022. [Online]. Available: <https://ko.wikipedia.org/wiki/%EC%9C%84%ED%82%A4%EB%B0%B1%EA%B3%BC:%EB%8C%80%EB%AC%B8>. [Accessed: 23-Nov-2022].

“[인공지능] Ann, DNN, CNN, RNN 개념과 차이,” 삶은 확률의 구름, 09-Oct-2021. [Online]. Available: <https://ebbnflow.tistory.com/119>. [Accessed: 23-Nov-2022].

<https://untitledtblog.tistory.com/154>

[http://www.incodom.kr/LSTM#h\\_cf1be63e24395cd2484630135ff89610](http://www.incodom.kr/LSTM#h_cf1be63e24395cd2484630135ff89610)

<https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e>  
<https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/>