

# Multimodal deep learning in text-to-image generation

-KT 딥러닝 세미나-

강사 김형욱  
(hukim@artiasolution.com)



# Contents

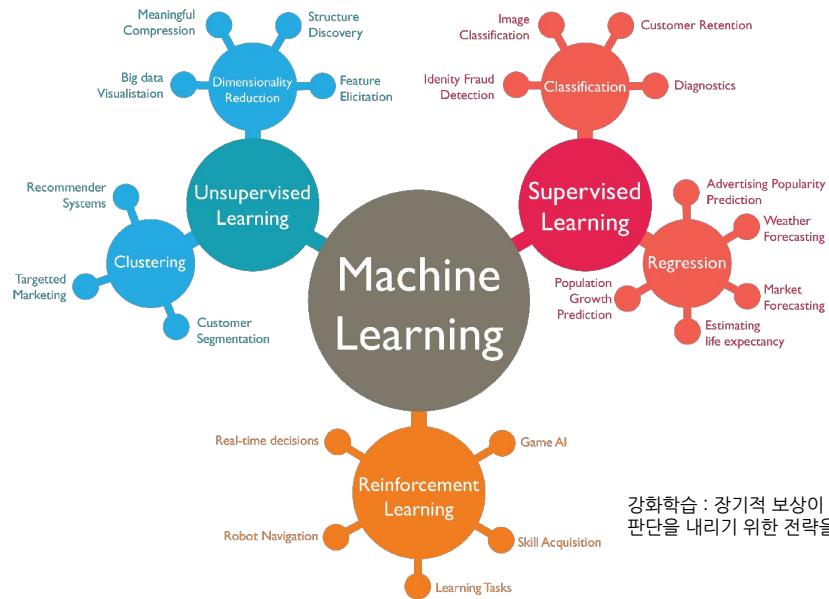
- Multimodal Deep learning (3 page)
- Recent trends in deep learning(19 page)
- Transformers(33 page)
- Self-supervision on vision tasks (66 page)
- Vision-language model pretraining (81 page)
- Deep generative learning (107 page)
- Diffusion models for text to image generation (128 page)

# Multimodal Deep Learning

- Introduction
- Applications

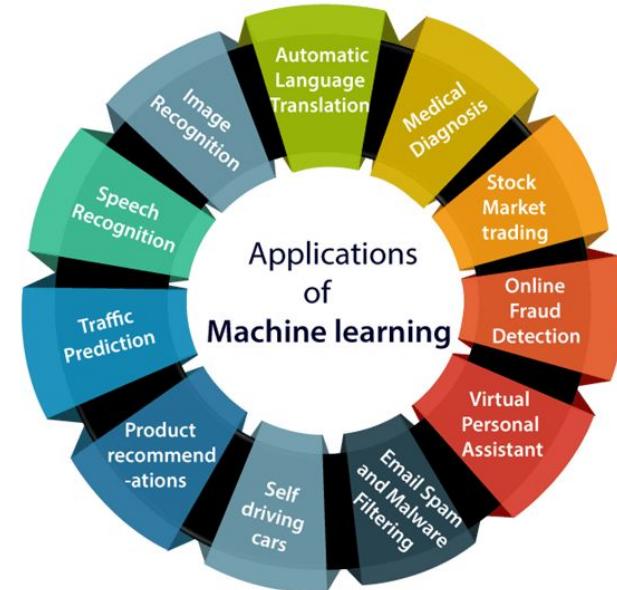
# AI는 어떻게 사용되고 있을까?

비지도학습 : 기계에게 데이터들의 분포, 구조를 탐색하게 하여 유용한 정보를 찾아낸다.



지도학습 : 기계에게 입력 데이터에 대해서 어떻게 예측할지 하나씩 정답을 알려주어 데이터-정답 사이 관계성을 학습한다.

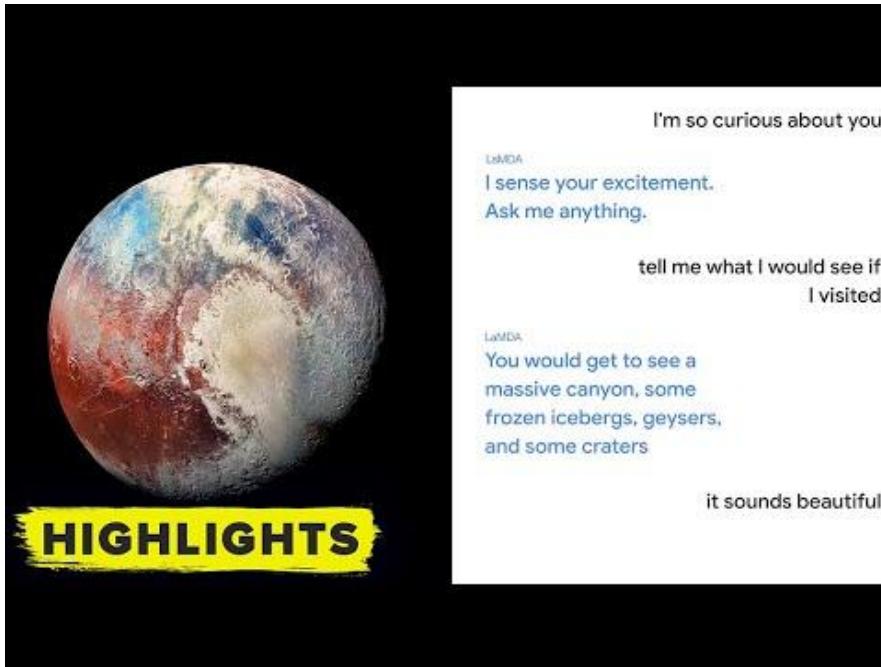
강화학습 : 장기적 보상이 가장 우수한 판단을 내리기 위한 전략을 학습한다.



〈머신러닝의 학술적 분류〉

〈머신러닝의 응용영역〉

# Google & DeepMind : various area

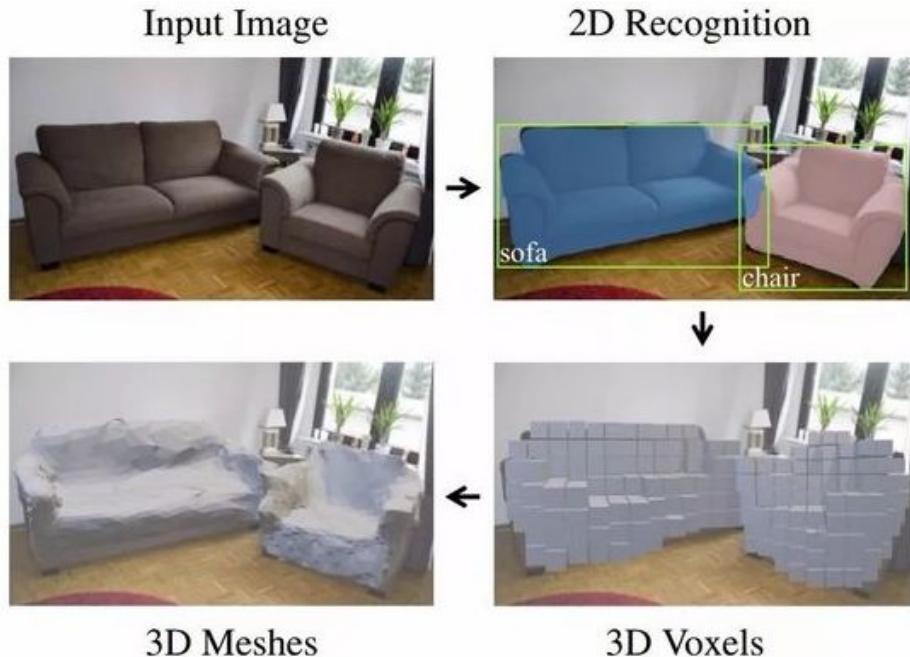


Google AI LaMDA

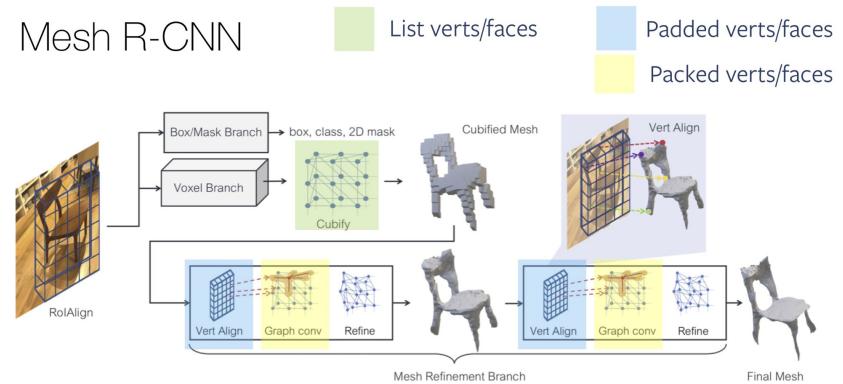


Google AI Imagen

# Meta AI : various area

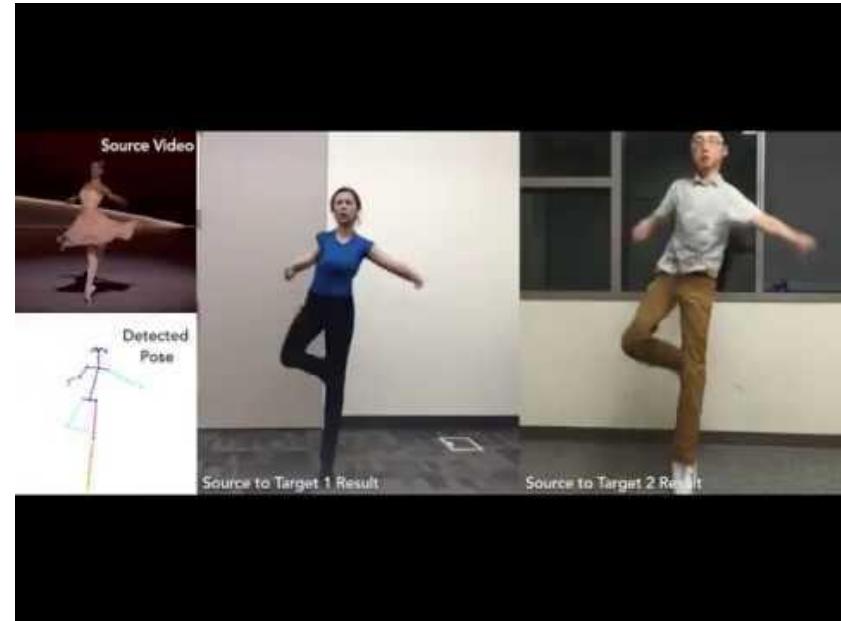
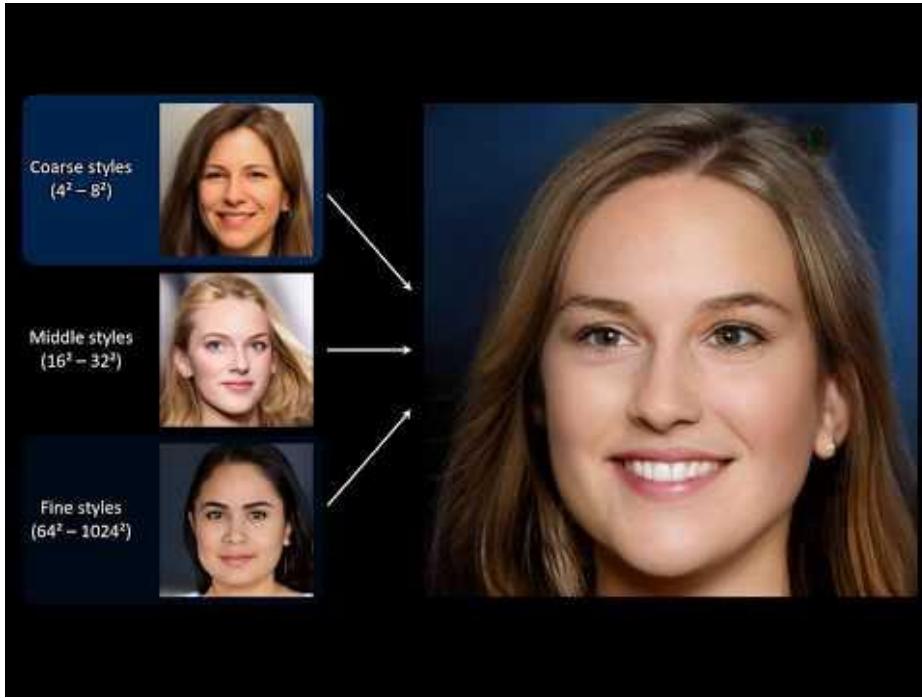


Mesh R-CNN



3D reconstruction

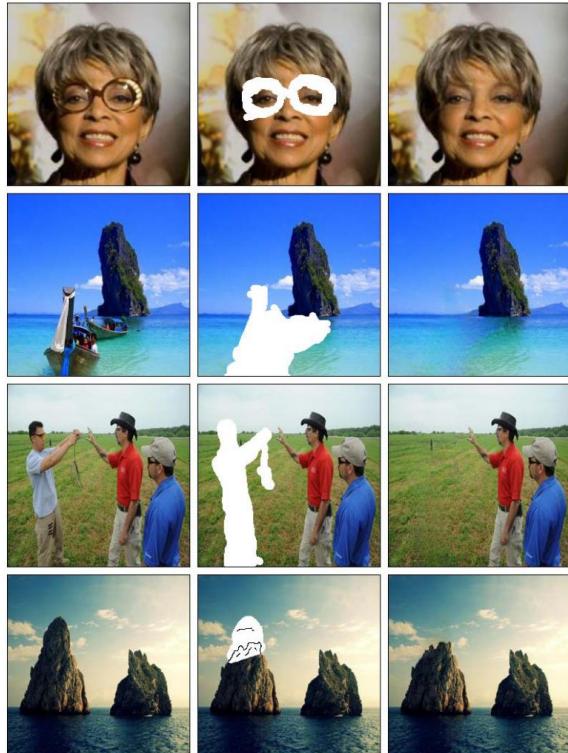
# Nvidia : Image generation



Everybody Dance Now

Style based GAN

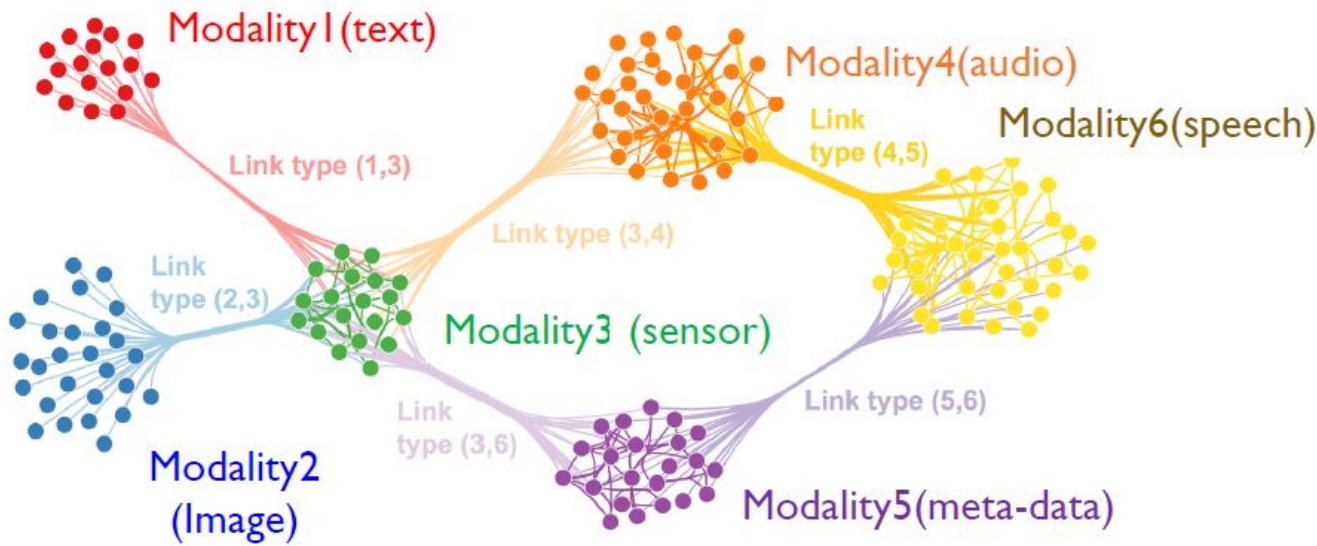
# Nvidia : Image completion



AI drawing

# Multimodal Deep Learning

Multimodal Deep Learning은 컴퓨터 비전, 자연어 처리, 음성 인식 등 다양한 모달리티(modality)에서 온 데이터를 처리하고 학습하는 딥러닝 기술



# Multimodal Deep Learning

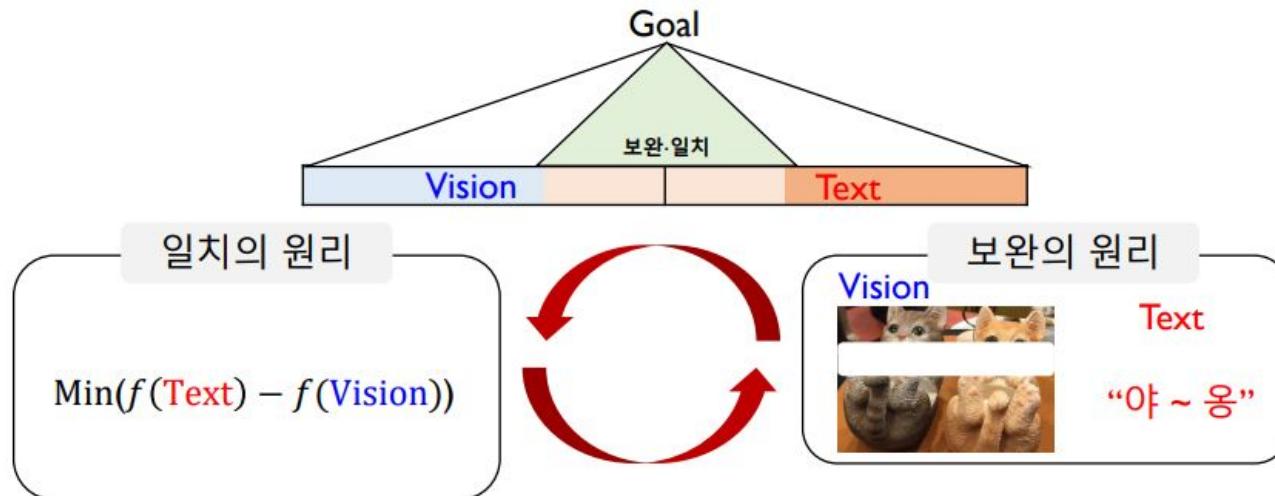
각 **modality**는 서로 다른 데이터 유형과 특성을 가지고 있기 때문에 이를 통합하여 하나의 시스템으로 학습하고 분석하는 것은 어려운 문제임.

$$\text{Model} = f(X_{term}^{doc}, X_{x,y}^{color}, X_{time}^{voice}, X_{time}^{sensor})$$

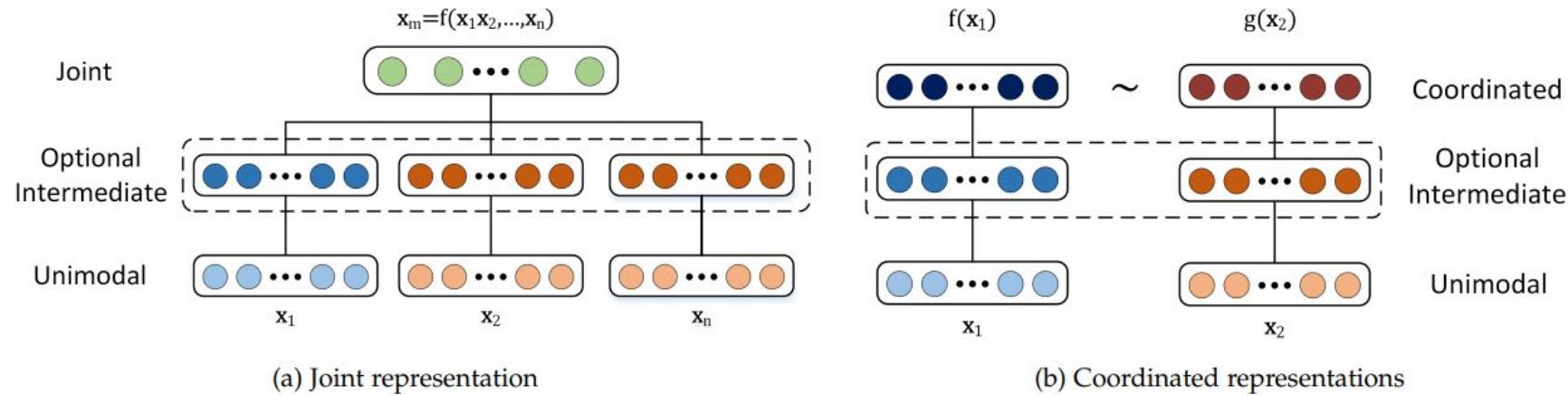
	Text	Image	...	Audio	Sensor	Y
관측치 1	...	...	...	...	...	0
관측치 2	...	...	...	...	...	1
...	...	...	...	...	...	...
관측치 N	...	...	...	...	...	0

# Multimodal Deep Learning의 목적

- 일치의 원리(consistency principle)
- 보완의 원리(complementary principle)

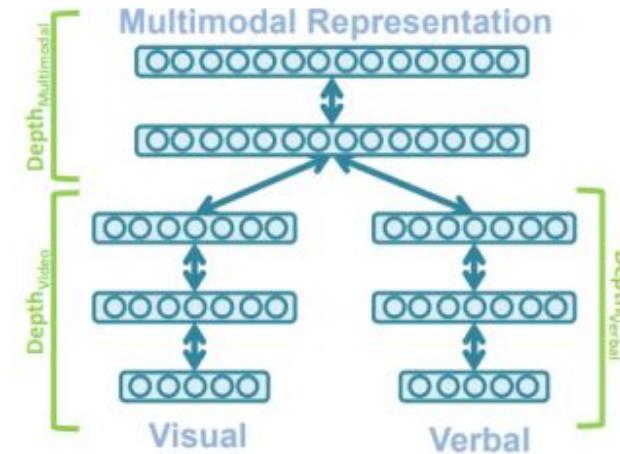
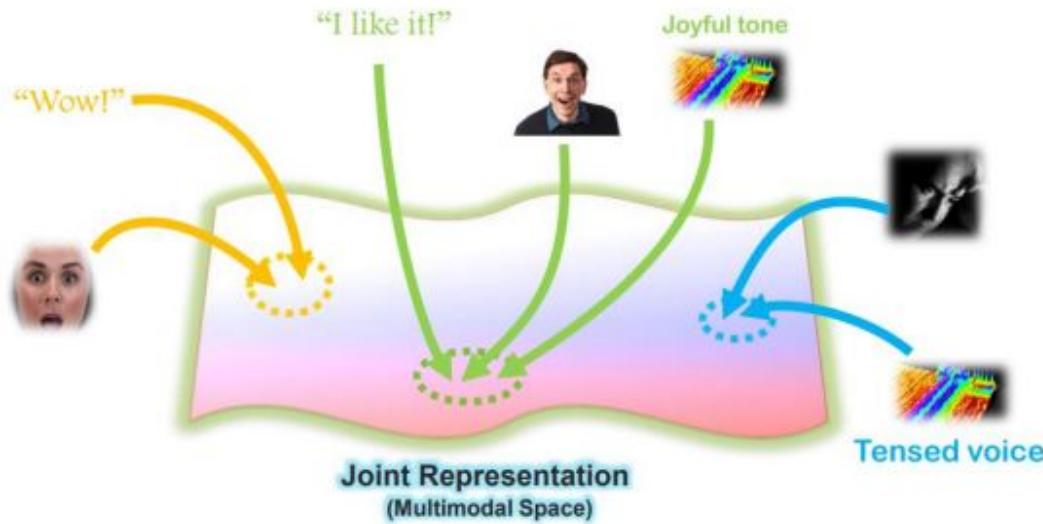


# Multimodal Deep Learning의 분류



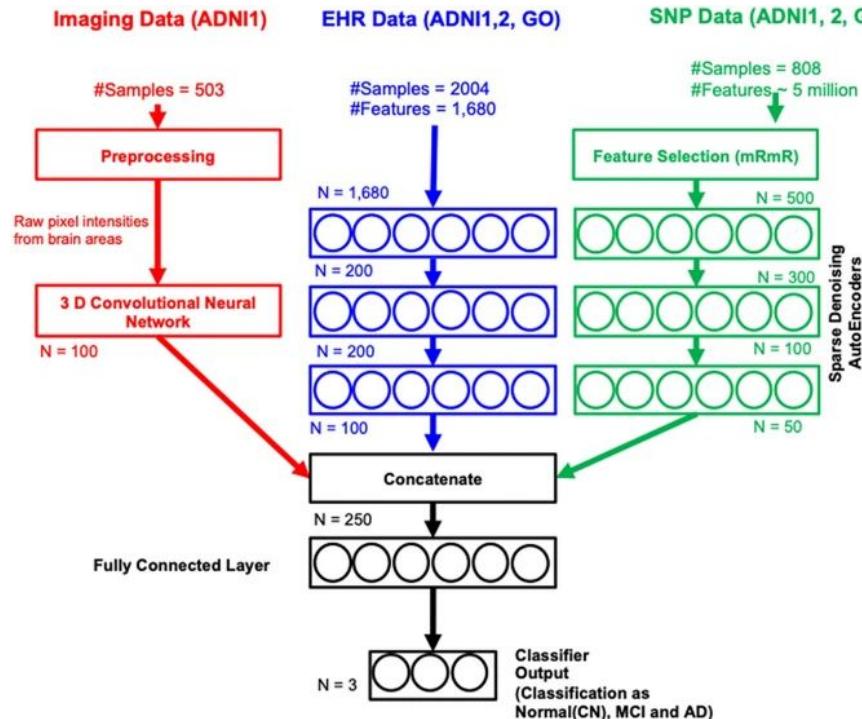
- **Joint Representation(feature fusion)** 방식은 각 모달리티의 입력을 하나의 특성 공간으로 통합하여 하나의 모델에서 동시에 학습하는 방식. 높은 기대성능이 장점
- **Coordinated Representation(simple concatenation)** 방식은 각 모달리티에서 추출한 특성을 각각 벡터로 변환한 뒤 하나의 벡터로 연결하여 하나의 모델에서 학습하는 방식. 특징이 어떤 데이터에서 추출한 것인지 알 수 있음.

# Multimodal Deep Learning의 분류



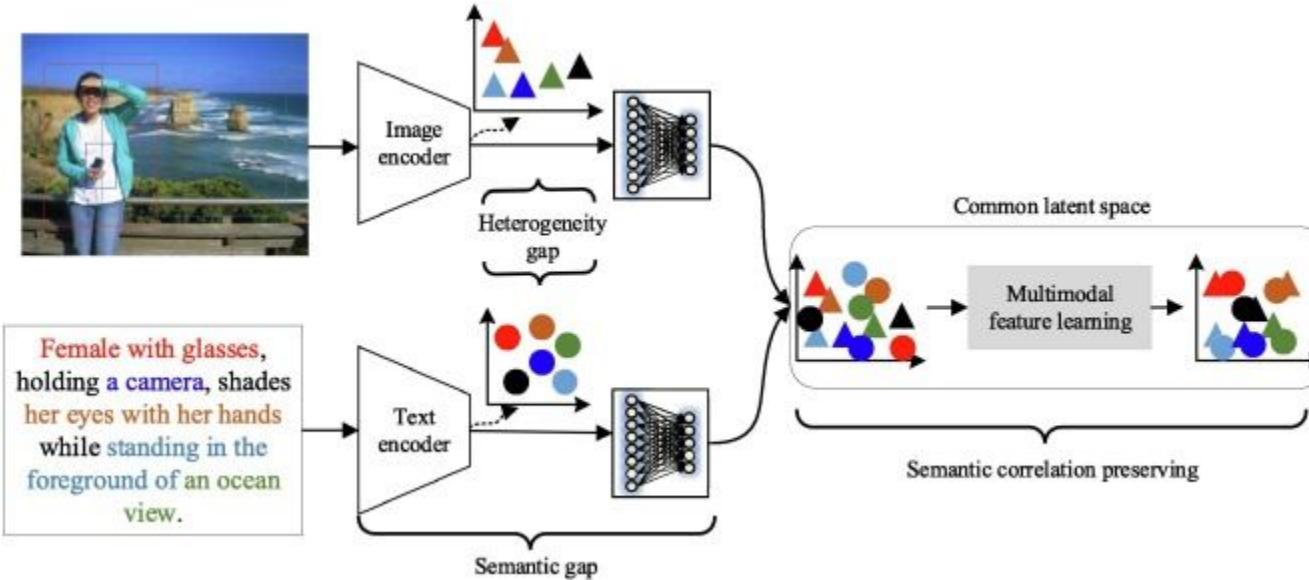
# Application : Multi-sensing for medical diagnosis

Multimodal deep learning models for early detection of Alzheimer's disease stage



# Application : Multimodal Content Understanding

feature fusion of image, text for multimodal representation



# Application : image captioning

Show, Attend and Tell(2015): Neural Image Caption Generation with Visual Attention

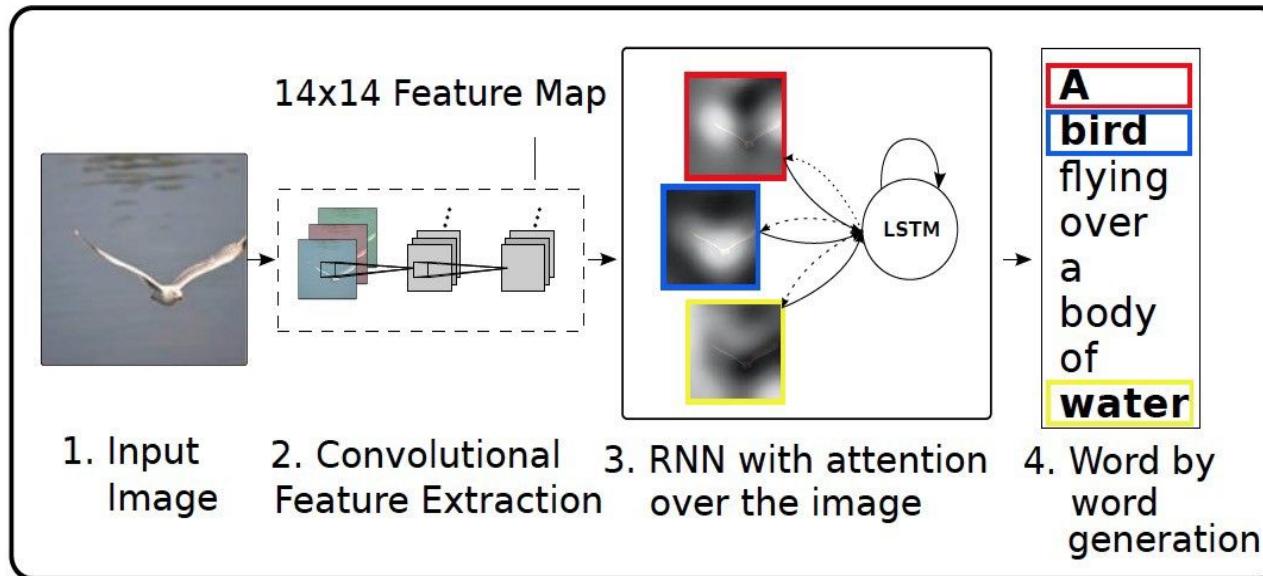
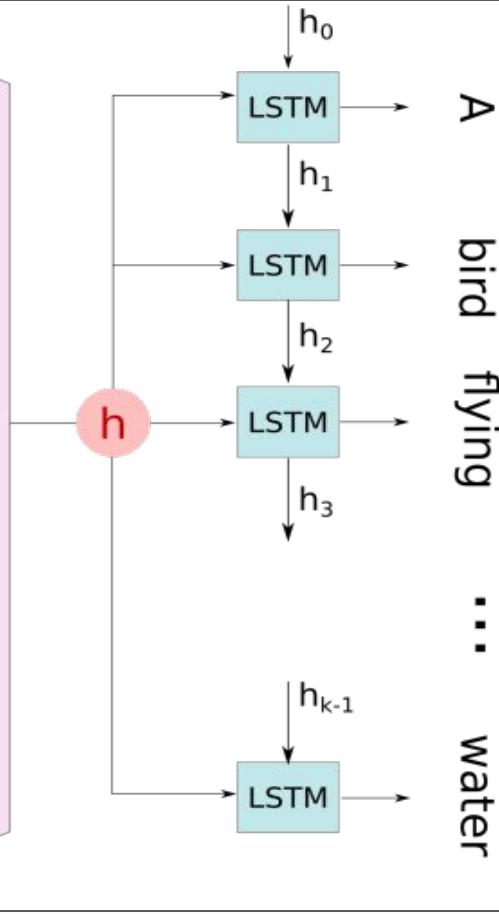


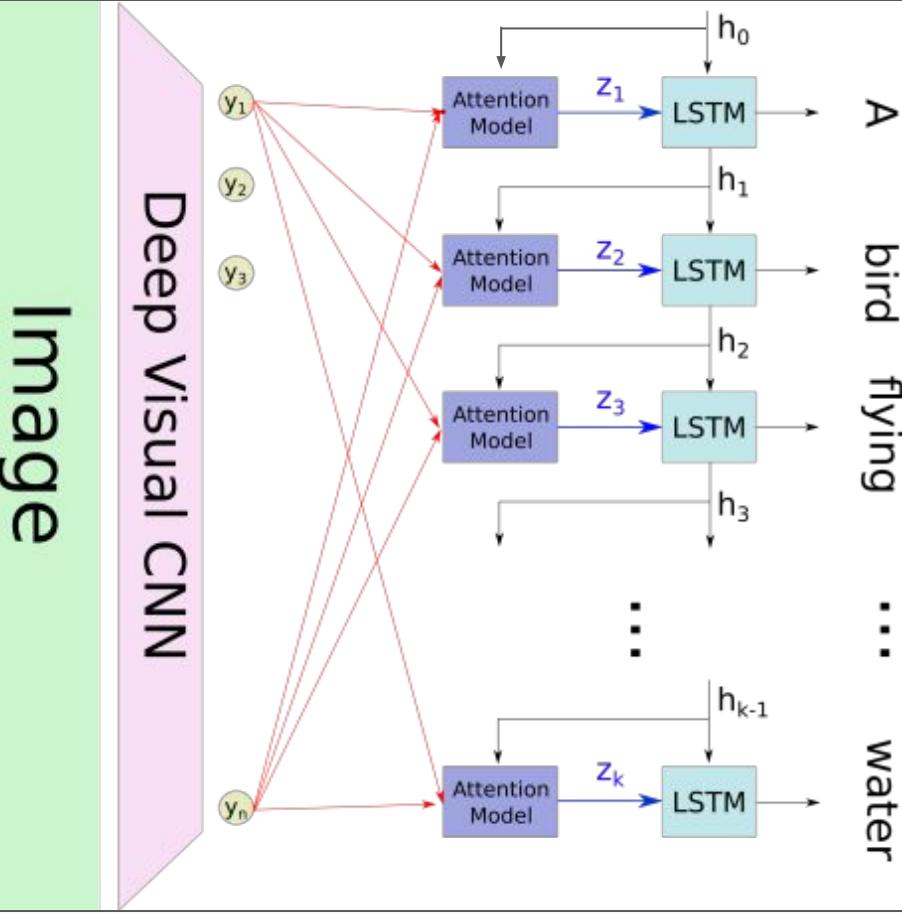
image  
captioning\*  
: encoder(CNN) +  
decoder(LSTM)

Image

Deep Visual CNN



<기존 image captioning>



<attention 적용>



A woman is throwing a frisbee in a park.



A dog is standing on a hardwood floor.



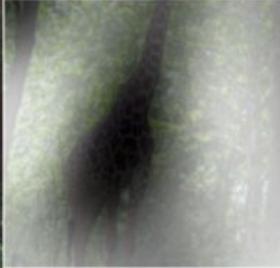
A stop sign is on a road with a mountain in the background.



A little girl sitting on a bed with a teddy bear.



A group of people sitting on a boat in the water.



A giraffe standing in a forest with trees in the background.

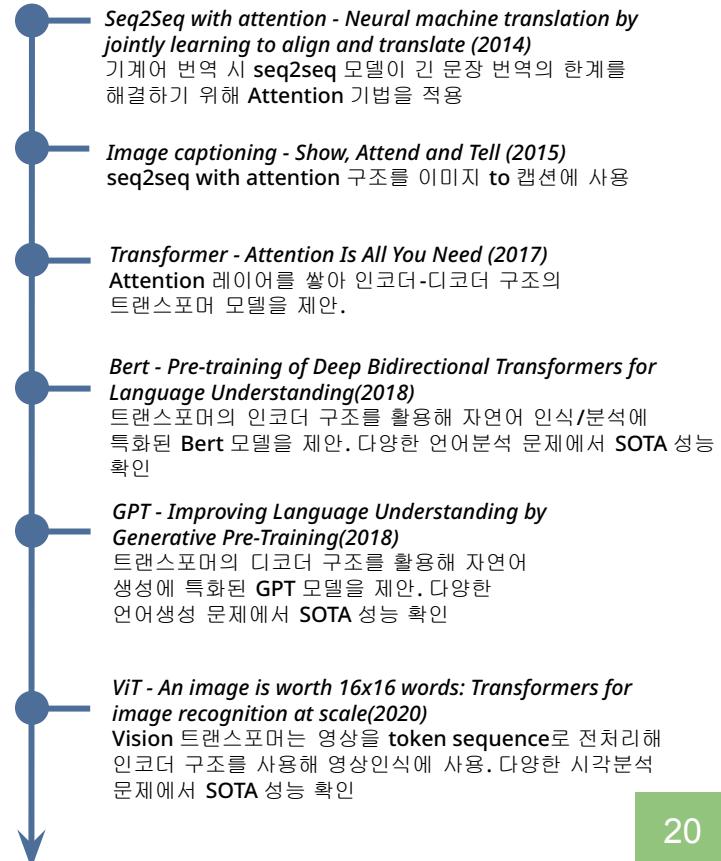
# Recent trends in deep learning

- Exploring the DL Trends
- See, Think, Confirm

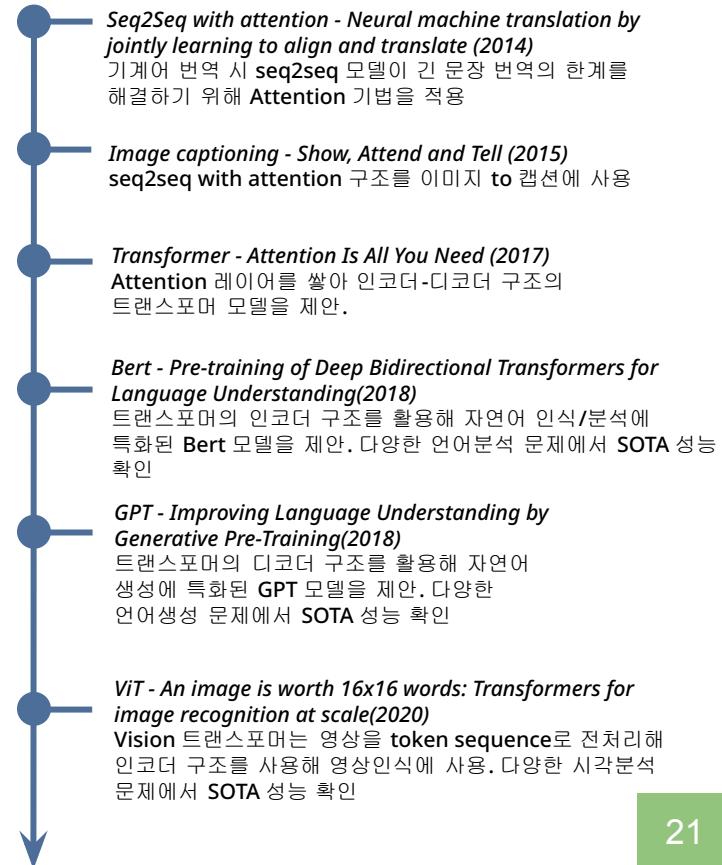
# Exploring the DL Trends

- (Transformer) 언어의 **context**를 이해하기 위해 사용된 **attention** 구조를 레이어로 쌓아 트랜스포머를 개발.
  - 언어문제의 주요 데이터셋에서 **SOTA** 성능 달성
  - 영상분석을 위한 비전트랜스포머는 이미지 분류, 객체 탐지, 이미지 분할 등 다양한 컴퓨터 비전 작업에서도 **SOTA** 성능 달성

왜 *Transformer*는 성능이 좋을까?



- (Transformer) CNN, RNN의 **inductive bias**는 소규모 데이터에서 효과적인 반면 트랜스포머는 일반성, 확장성, 강건성에서 우월함.
  - **Self-supervision** 발전으로 학습 데이터의 규모가 크게 확장되었음. 또한 **masked language modeling**처럼 **task-agnostic learning**이 가능해져서 일반성도 좋음.
  - 확장성을 가지고 대규모 모델을 사전학습 후 전이학습하는 것이 더 효과적임. 전이학습은 또한 환경친화적임.
  - 모델이 크면 오히려 과적합과 데이터 노이즈에 더 강건하다는 최신 연구들이 존재함.

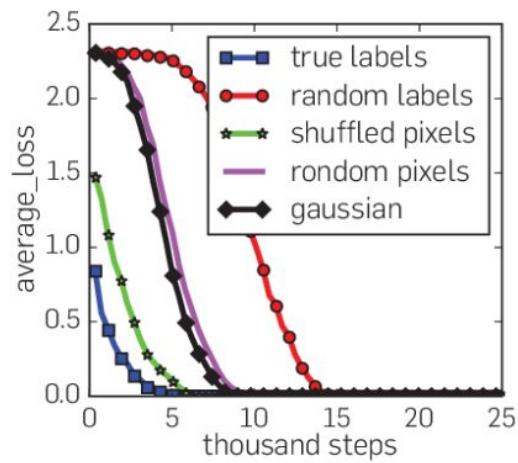


- Chiyuan zhang(2017)은 학습데이터의 레이블을 무작위로 섞어도 뉴럴넷의 학습에러가 잘 감소되는 현상 발견
  - 이미지 픽셀을 완전히 랜덤하게 변경해도 동일하게 잘 학습됨.
  - 즉 뉴럴넷이 어떤 인지적 특징을 학습하는 것이 아니라 사진 데이터를 통째로 암기한다는 의미.
  - 모델의 복잡도가 높지 않아도 비슷한 현상이 나타남.



Chiyuan zhang(MIT)

학습데이터의 레이블을 무작위로 섞어버려도 학습이 잘 되는데요? CNN이 사진을 그냥 통째로 외워버리는 것 같아요.



- Devansh Arpit(2017)은 연구를 통해 적정 모델의 복잡도는 데이터의 수량이 아닌 패턴의 난이도로 결정해야 한다고 보고.
  - 뉴럴넷은 학습 초기에 파악하기 쉬운 패턴을 학습하고 진행될수록 어려운 패턴을 학습함.
  - 따라서 마지막에 데이터를 암기하는 것처럼 보이는 것.
  - 모델의 용량이 높을 수록 노이즈 샘플에 의한 학습에 강건해짐.



Devansh Arpit  
(Université de Montréal)

자세히 뜯어보니 시간에 따라 쉬운 패턴에서 복잡한 패턴을 학습함. 그래서 모델의 적정용량은 데이터 수량이 아니라 패턴의 복잡도에 상관성을 가져야 함.

근데 재밌는게 모델의 용량이 높으면 노이즈를 섞어 학습해도 헛깔리지 않고 정상적으로 잘 추론하는 능력이 좋아짐.

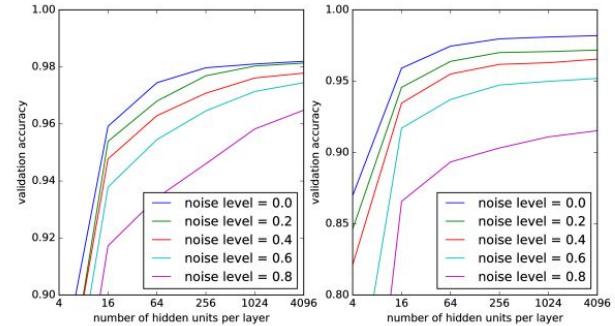


Figure 5. Performance as a function of capacity in 2-layer MLPs trained on (noisy versions of) MNIST. For real data, performance is already very close to maximal with 4096 hidden units, but when there is noise in the dataset, higher capacity is needed.

- Preetum Nakkiran(2019)은 double descent 현상을 발견함.

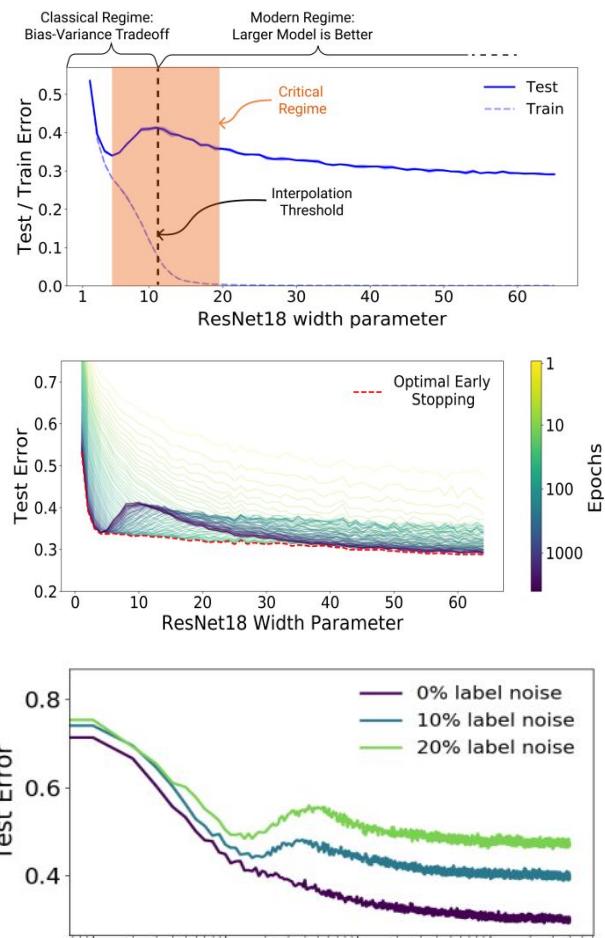
- 모델의 복잡도를 계속 올리다보면 validation error가 다시 감소하는 구간이 존재함.
- 시간축에서도 학습을 오래하면 다시 성능이 개선되는 double descent 구간이 존재.



Preetum Nakkiran(Harvard univ.)

형들, 검증에러가 감소하다가 갑자기 증가하면 오버피팅이라고 했는데 오히려 모델사이즈랑 학습시간이 커질수록 다시 감소하는 구간(double descent) 발견했어요!

기존 과적합 이론하고 얼리스탑하는 방식이 틀렸고 모델을 더 복잡하게 해서 더 오래 학습하면 성능이 계속 좋아질수도 있겠는데요?!



Nakkiran, Preetum, et al. "Deep Double Descent: Where Bigger Models and More Data Hurt." arXiv preprint arXiv:1912.02292 (2019).  
 Nakkiran, Preetum, et al. "Deep double descent: Where bigger models and more data hurt." Journal of Statistical Mechanics: Theory and Experiment 2021.12 (2021): 124003.

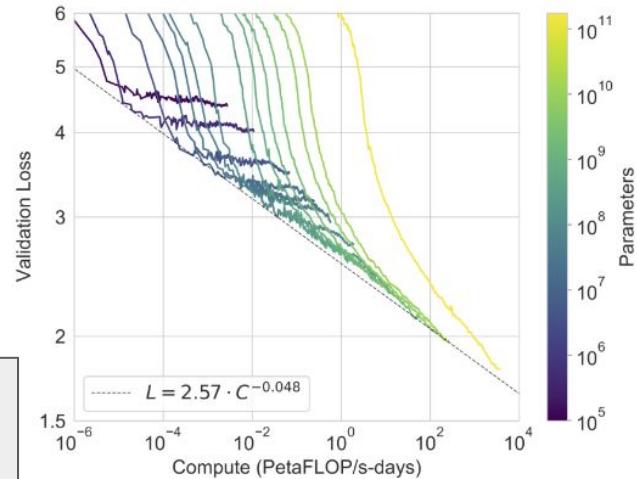
- Tom B. Brown(2020)은 GPT-3 모델의 **few-shot learner**

연구를 진행하면서 파라미터가 많은 수록 더 높은 성능과 연산량 대비 더 빠른 학습 속도를 보임을 보고함.



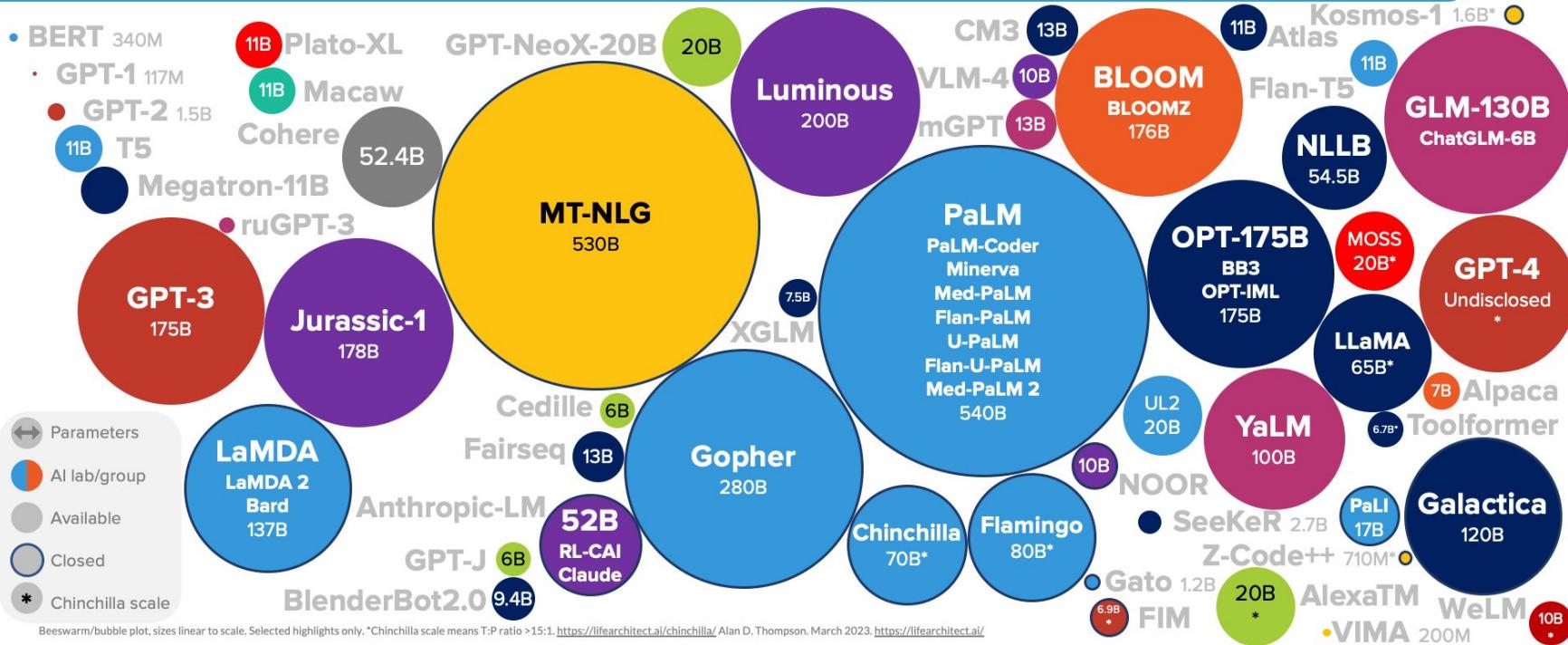
내가 실제로 MS에서 투자 받은 돈으로 모델  
파라미터를 몇십 억개 짜리가 아니라  
1750억개로 GPT 만들어서 flex해보니까 모델이  
복잡하면 복잡할수록 더 좋을뿐만 아니라 학습  
속도도 더 빨라짐 ㅋㅋ

그러니까 모델은 크면 클수록 좋다는 결론임!



Tom B. Brown(OpenAI)

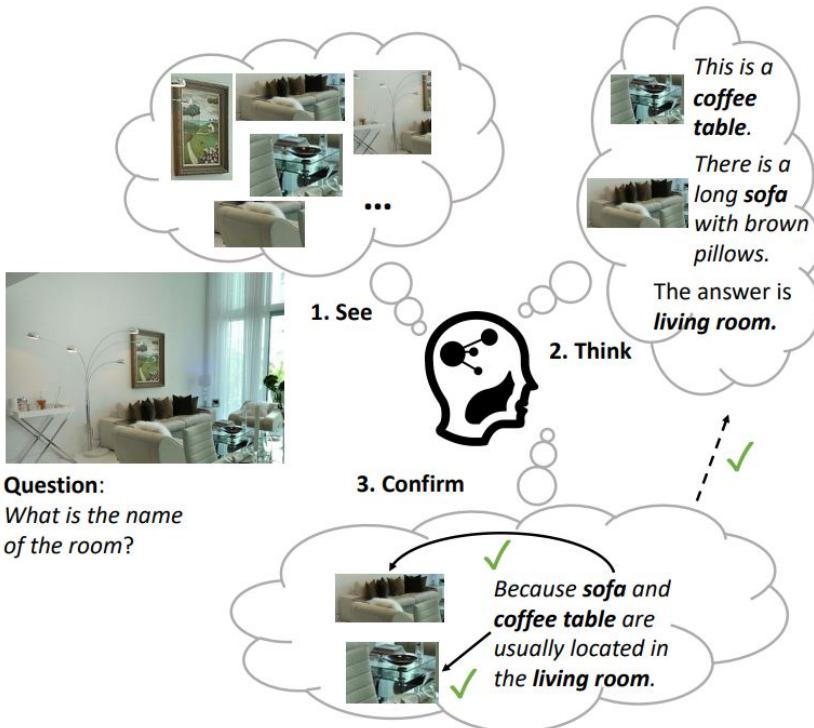
# LANGUAGE MODEL SIZES TO MAR/2023



LifeArchitect.ai/models

- (**Self-supervised learning, SSL**) **Transformer** 기반 모델은 성능의 상한이 높지만 방대한 학습 데이터셋 필요. 이에 따라 **SSL**은 딥러닝의 매우 중요한 또 다른 트렌드임.
  - 지도학습은 사람에 의한 **Golden label**이 필요, 반면 **SSL**은 레이블 없이 데이터를 학습 가능. 따라서 **SSL**은 레이블링 비용을 대폭 절감해 초대규모 데이터셋에서 학습 가능함.
- (**Vision-language modeling**) 최근 언어와 시각을 결합하여 이미지와 텍스트 간의 상호작용을 고려한 **CLIP** 모델은 중요한 **DL** 트렌드임.
  - **CLIP**은 이미지와 텍스트 사이의 상호작용을 고려해 생성모델의 프롬프트 엔지니어링 등 다양한 응용분야에서 활용.
- (**textual prompting**) **CLIP** 이후 언어가 아닌 다른 데이터를 언어정보인 **textual prompt**로 변환하는 방식이 주목 받음. 언어화된 멀티모달 데이터들을 언어모델인 **GPT**를 사용해 분석, 해결하는 방식으로 통합 AI시스템을 구축할 수 있음. 따라서 **textual prompting**은 멀티모달 통합, **AGI(Artificial General Intelligence)**과 같은 강인공지능 연구의 핵심 키워드.

# See, Think, Confirm



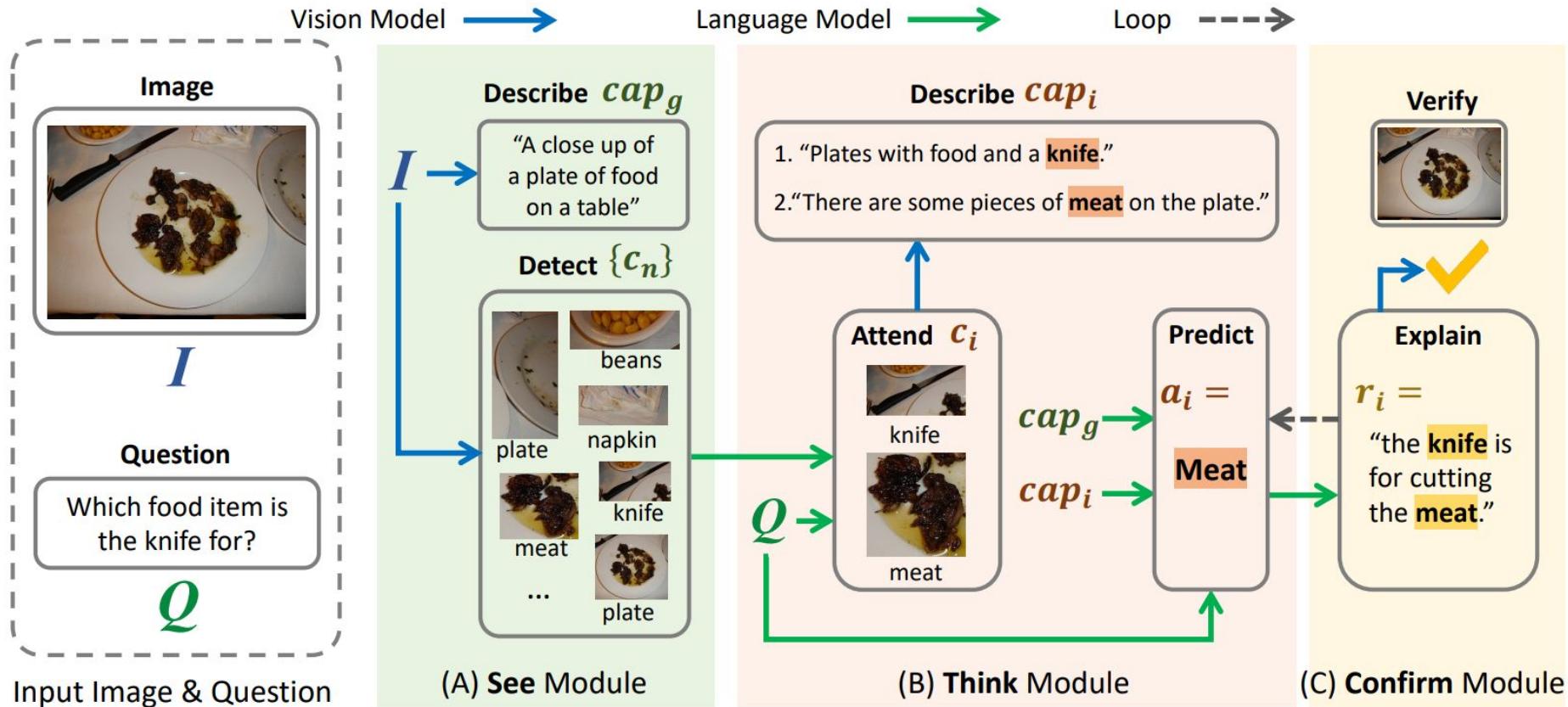
<Interactive Prompting Visual Reasoner (IPVR)>

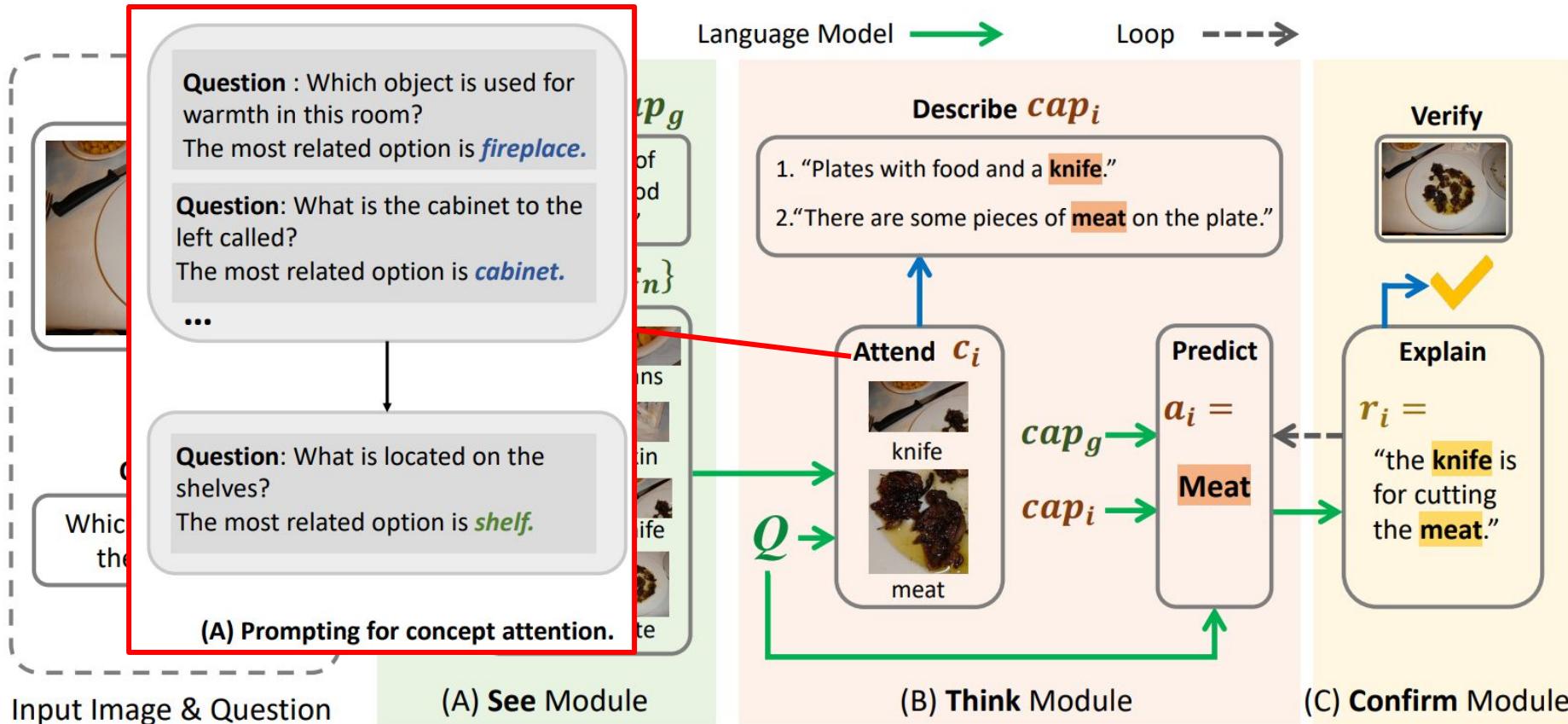
**See :** 이미지에서 가능한 모든 **visual concept**를 추출.  
(extract - RCNN으로 객체검출.)

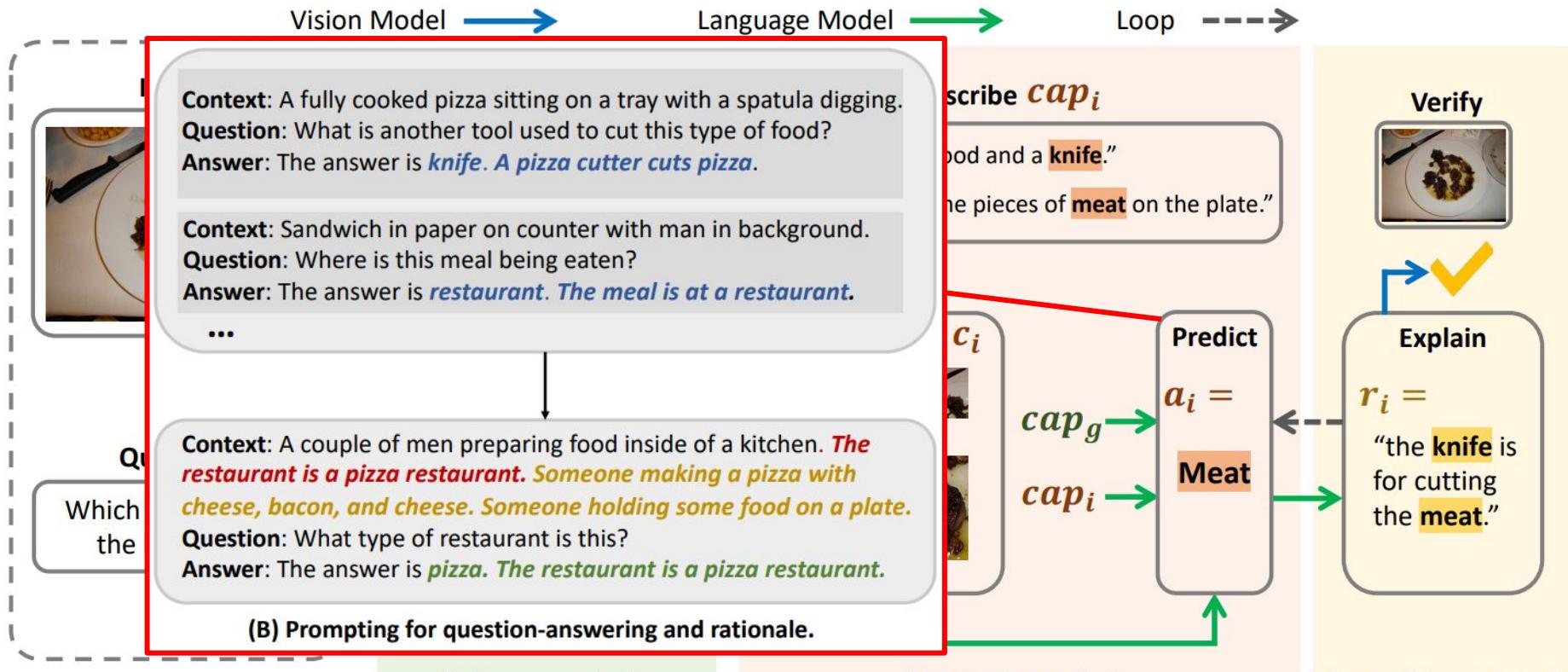
**Think :** LLM을 사용해 **visual concept** 중 질문에  
연관된 것을 캡셔닝해 문제에 답변  
(attend - LLM, describe - image captioner, predict -

**Confirm :** 논리적 검증을 통해 투명하고 신뢰성 있는  
추론을 제공.

Chen, Zhenfang, et al. "See, Think, Confirm: Interactive Prompting Between Vision and Language Models for Knowledge-based Visual Reasoning." arXiv preprint arXiv:2301.05226 (2023).







## Input Image & Question

(A) See Module

## (B) Think Module

### (C) Confirm Module

	Input	Global Caption	Attend	Explain	Prediction	
Image		Caption: A man running across a tennis court holding a racquet.	Ball: Someone has a tennis racket and is about to hit a ball. Tennis court: Someone playing a sport on a tennis court. All tags: fence, ball, car, house, man, roof, bat, tennis court, shirt, window...	The fence is meant to block the ball. The fence is meant to block cars.	ball ball ✓ cars ✗ cars ✗	Ours CoT PiCa
Question	What is the fence meant to block?					
GT	ball					
Image		Caption: A living room with a couch a table and a lamp.	Picture: Someone is taking a picture of some art. Picture: There's a lamp and a picture hanging on the wall. All tags: lamp, wall, door, couch, pillow, pillow, picture, picture, table...	The wall is used for displaying art. The wall is used for a sofa.	art art ✓ sofa ✗ wall ✗	Ours CoT PiCa
Question	What is this wall used for?					
GT	art					

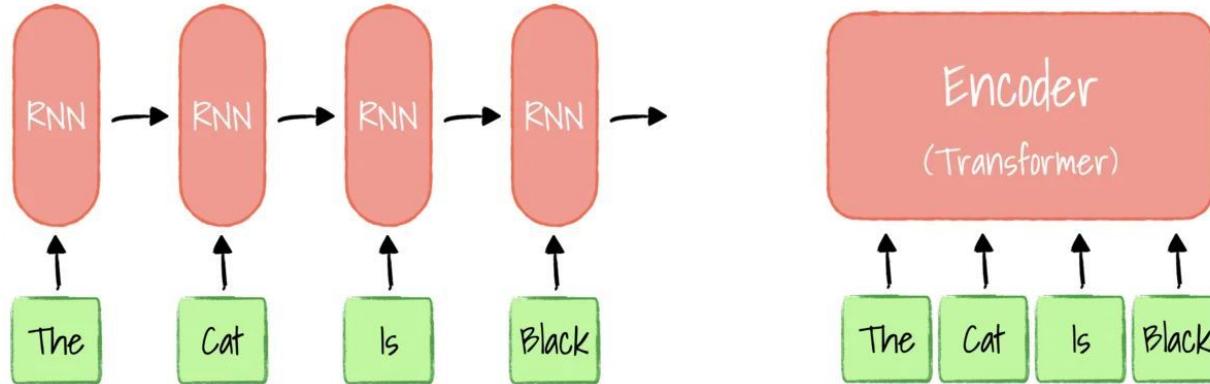
Figure 4. Qualitative results of our IPVR and baselines. Besides better answer accuracy, our method also enjoys better transparency by providing a step-by-step reasoning trace with related visual concepts, regional descriptions, and a supporting rationale. → denotes our reasoning flow.

# Transformer

- Transformer
- Bert
- GPT
- Vision Transformer

# Transformer

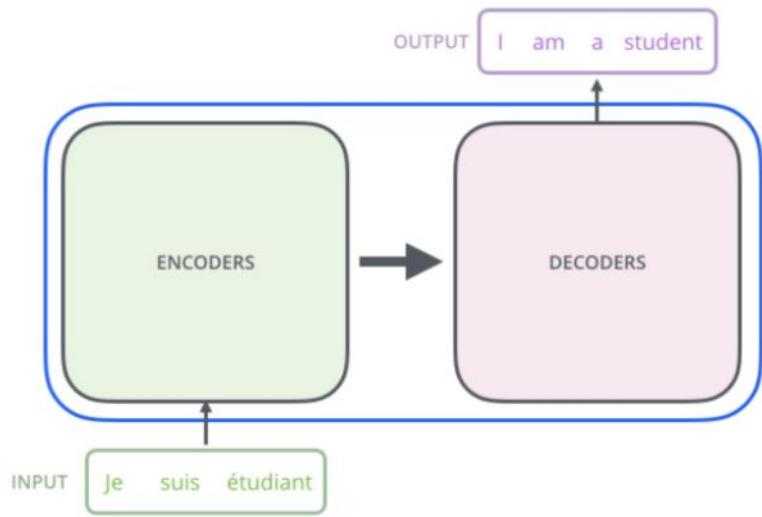
- NIPS 2017에서 Google은 “Attention Is All You Need” 논문에서 RNN을 대체하는 Transformer 구조 발표
  - RNN의 단점은 순열 데이터의 time-step을 순차적으로 인코딩  
➤ 연산시간 증가
  - Transformer는 순열 데이터의 time-step을 병렬적으로 인코딩  
➤ 연산시간 감소



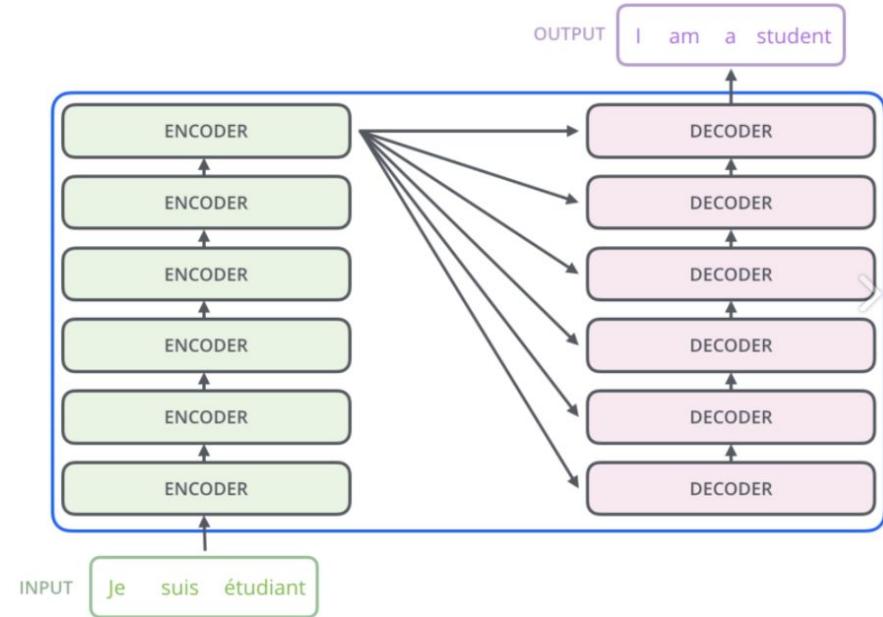
# Transformer

- **Machine translation**에서 사용되는 **Seq2Seq** 모델의 **attention mechanism**을 명시적 신경망 구조로 구현하는 것이 기본 아이디어.
- **Transformer** 기반 언어모델링은 **NLP**의 다양한 문제들에서 다른 언어모델 대비 가장 우수한 성능을 보임.
- 대표적으로 언어모델로 **GPT**와 **Bert** 계열이 있고, 시각모델로 **ViT**를 사용.

# Transformer : Encoder-Decoder

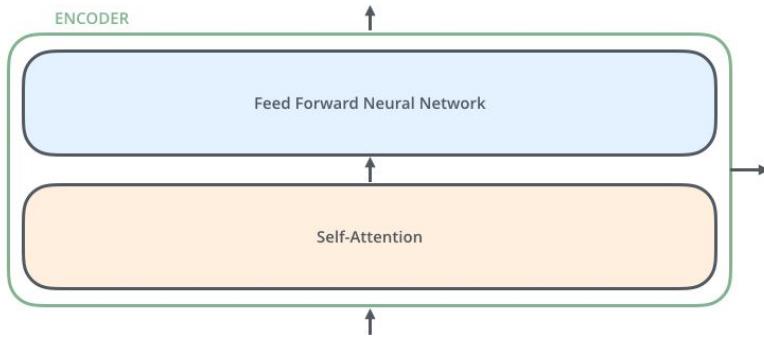


<Transformer의 구조 :  
Encoder-Decoder>



<Encoder-Decoder의 내부>

# Transformer : Encoder



- Encoder는 모두 똑같은 구조
- 두 개의 sub-layer로 구성

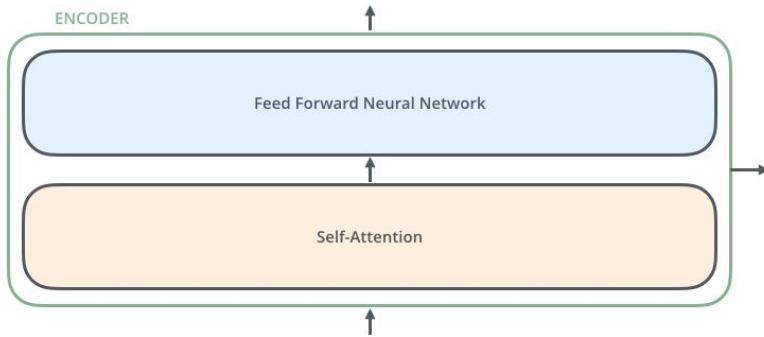
## 1. Self-Attention

하나의 단어를 encode할 때, 입력 내 모든 단어들과 관계 (Attention)를 고려.

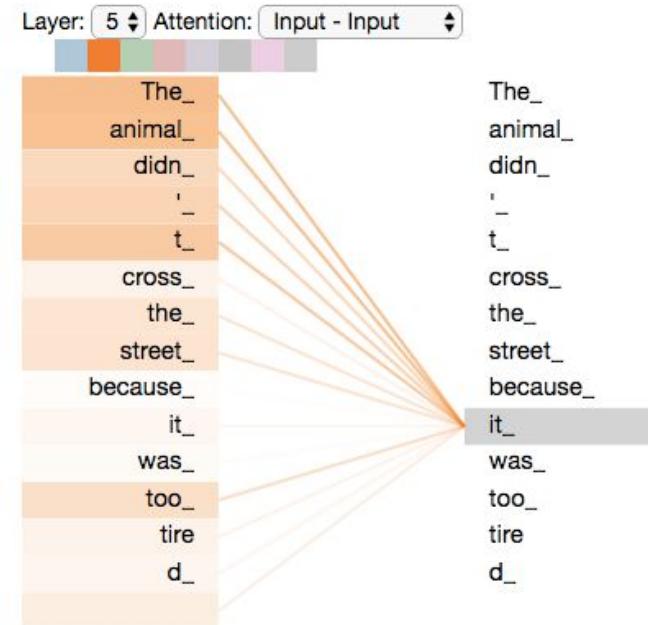
## 2. Feed Forward Neural Network

- Self-Attention 층을 통과한 출력은 feed-forward 신경망으로 입력.
- FFNN은 fully-connected layer로 구성,
- 각 단어에 대한 독립적인 출력 계산.

# Transformer : Encoder

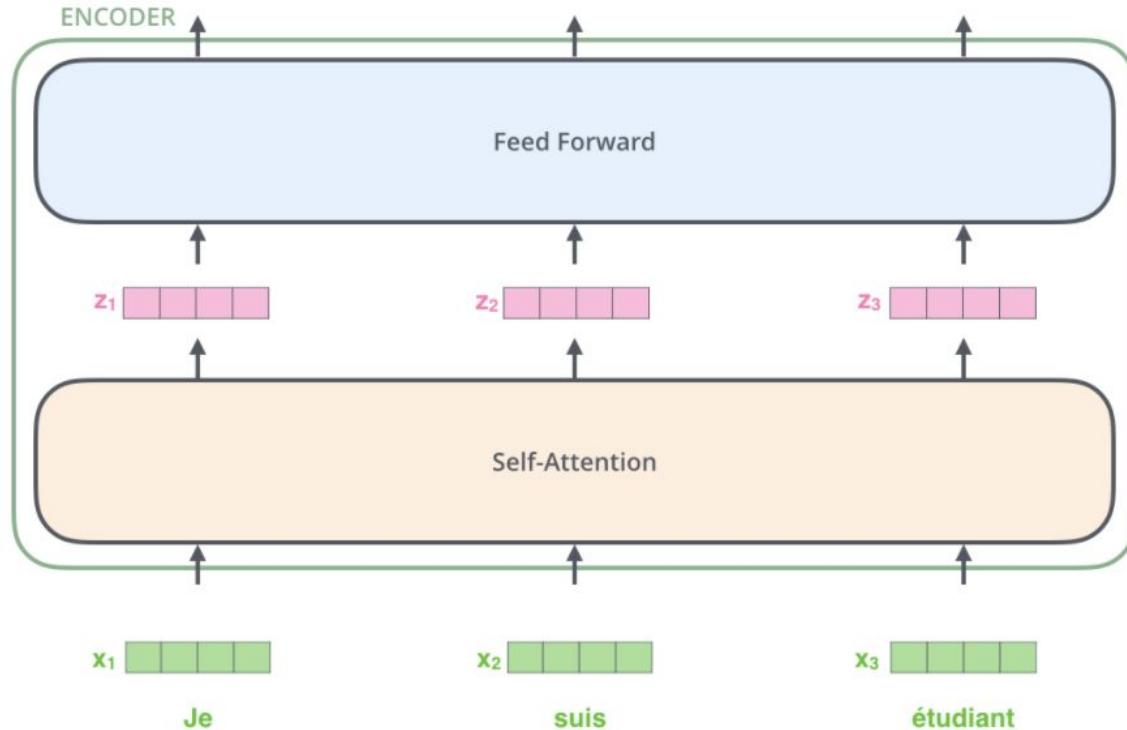


- Encoder는 모두 똑같은 구조
- 두 개의 sub-layer로 구성



Attention 효과 : 문장에서 관계대명사가  
지칭하는 단어추론

# Transformer : Encoder



## 〈FFNN〉

출력 : 64차원 벡터의 리스트

- Self-Attention의 출력은 독립적으로 병렬처리
- 벡터의 차원은 하이퍼파라미터

## 〈Self-Attention〉

출력 : 64차원 벡터의 리스트

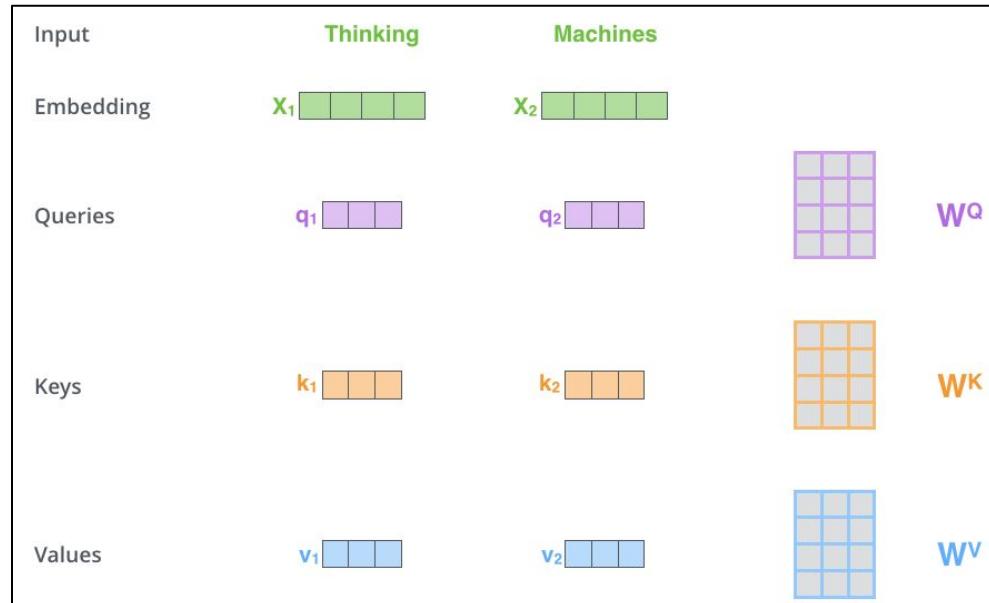
- 리스트의 크기는 문장 길이
- 벡터의 차원은 하이퍼파라미터

## 〈Embedding layer〉

출력 : 512차원 벡터의 리스트

- 리스트의 크기는 문장 길이
- 벡터의 차원은 하이퍼파라미터

# *Self-Attention in Encoder(1)*



<Self-Attention을 위한 Query, Key, Value 개념>

- [‘Thinking’, ‘Machine’] 시퀀스의 입력 가정
- Encoder의 입력된 벡터를 학습가능한 3개의 행렬과 곱해, Query, Key, Value 벡터 생성.
- 모두 embedding vector 보다 낮은 차원 (64차원)의 벡터 사용.
- 각 벡터는 attention을 계산하기 위해 도입한 추상적 개념을 의미.

**<Query>**

- 입력 단어와 다른 단어의 관계를 질의하는 벡터

**<Key>**

- 다른 단어가 입력 단어 사이의 관계를 질의 받는 벡터

**<Value>**

- 입력 단어에 대한 의미 정보를 내포한 벡터

# *Self-Attention in Encoder(2)*

Input	Thinking		Machines	
Embedding	$x_1$	[4 green boxes]	$x_2$	[4 green boxes]
Queries	$q_1$	[3 purple boxes]	$q_2$	[3 purple boxes]
Keys	$k_1$	[3 orange boxes]	$k_2$	[3 orange boxes]
Values	$v_1$	[3 blue boxes]	$v_2$	[3 blue boxes]
Score	$q_1 \cdot k_1 = 112$		$q_1 \cdot k_2 = 96$	



Input	Thinking		Machines	
Embedding	$x_1$	[4 green boxes]	$x_2$	[4 green boxes]
Queries	$q_1$	[3 purple boxes]	$q_2$	[3 purple boxes]
Keys	$k_1$	[3 orange boxes]	$k_2$	[3 orange boxes]
Values	$v_1$	[3 blue boxes]	$v_2$	[3 blue boxes]
Score	$q_1 \cdot k_1 = 112$		$q_1 \cdot k_2 = 96$	
Divide by 8 ( $\sqrt{d_k}$ )	14		12	
Softmax	0.88		0.12	

<Query과 Key의 내적을 통한  
attention score 계산>

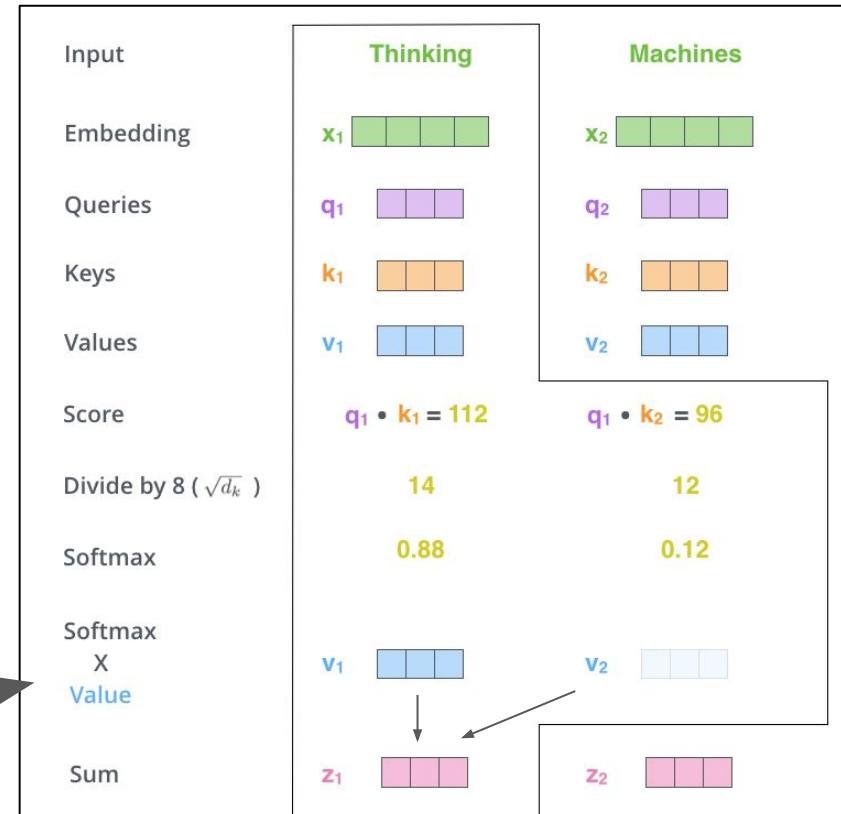
<Softmax를 사용한  
attention distribution 계산>

# Self-Attention in Encoder(3)

Input	Thinking	
Embedding	$x_1$ [green green green green]	
Queries	$q_1$ [purple purple purple]	
Keys	$k_1$ [orange orange orange]	
Values	$v_1$ [blue blue blue]	
Score	$q_1 \cdot k_1 = 112$	
Divide by 8 ( $\sqrt{d_k}$ )	14	
Softmax	0.88	
Input	Machines	
Embedding	$x_2$ [green green green green]	
Queries	$q_2$ [purple purple purple]	
Keys	$k_2$ [orange orange orange]	
Values	$v_2$ [blue blue blue]	
Score	$q_1 \cdot k_2 = 96$	
Divide by 8 ( $\sqrt{d_k}$ )	12	
Softmax	0.12	



attention distribution을 사용한,  
Value의 weighted sum



# *Self-Attention* 과정을 *matrix* 형태로 표현

$$\begin{array}{c} X \quad W^Q \quad Q \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \end{array}$$
$$\times =$$
$$\begin{array}{c} X \quad W^K \quad K \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \end{array}$$
$$\times =$$
$$\begin{array}{c} X \quad W^V \quad V \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \\ \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \quad \begin{matrix} \text{---} \\ | \end{matrix} \end{array}$$

〈Embeddings에 대해  
Queries, Keys, Values 계산〉



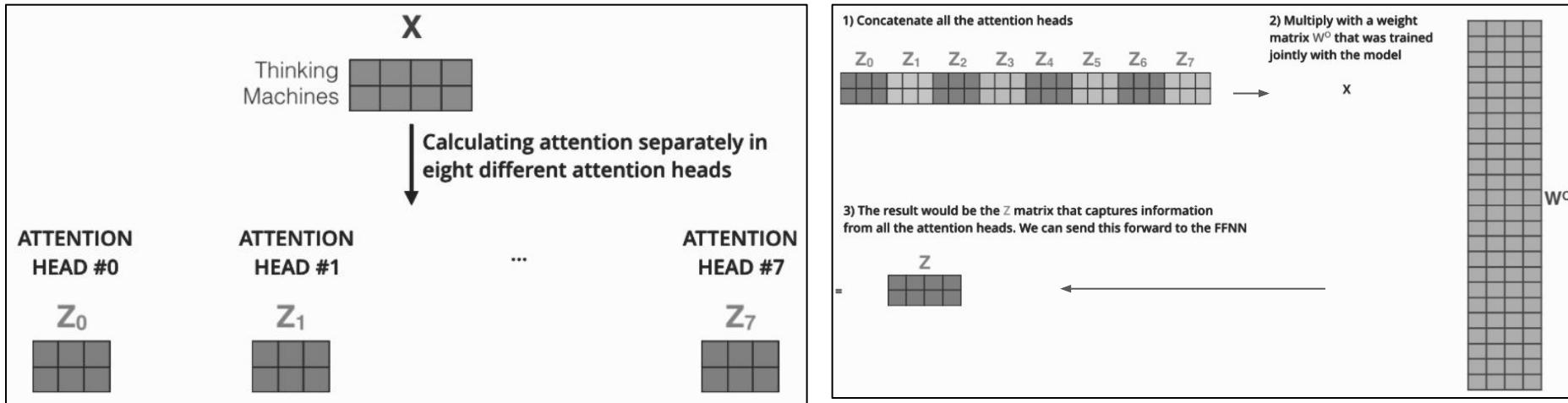
$$\text{softmax} \left( \frac{Q \times K^T}{\sqrt{d_k}} \right) \cdot V$$
$$= Z$$

각 입력의 Value 벡터의 가중치합이 출력

〈행렬곱 형태로 Self-Attention 계산〉

# Multi-head self-attention(1)

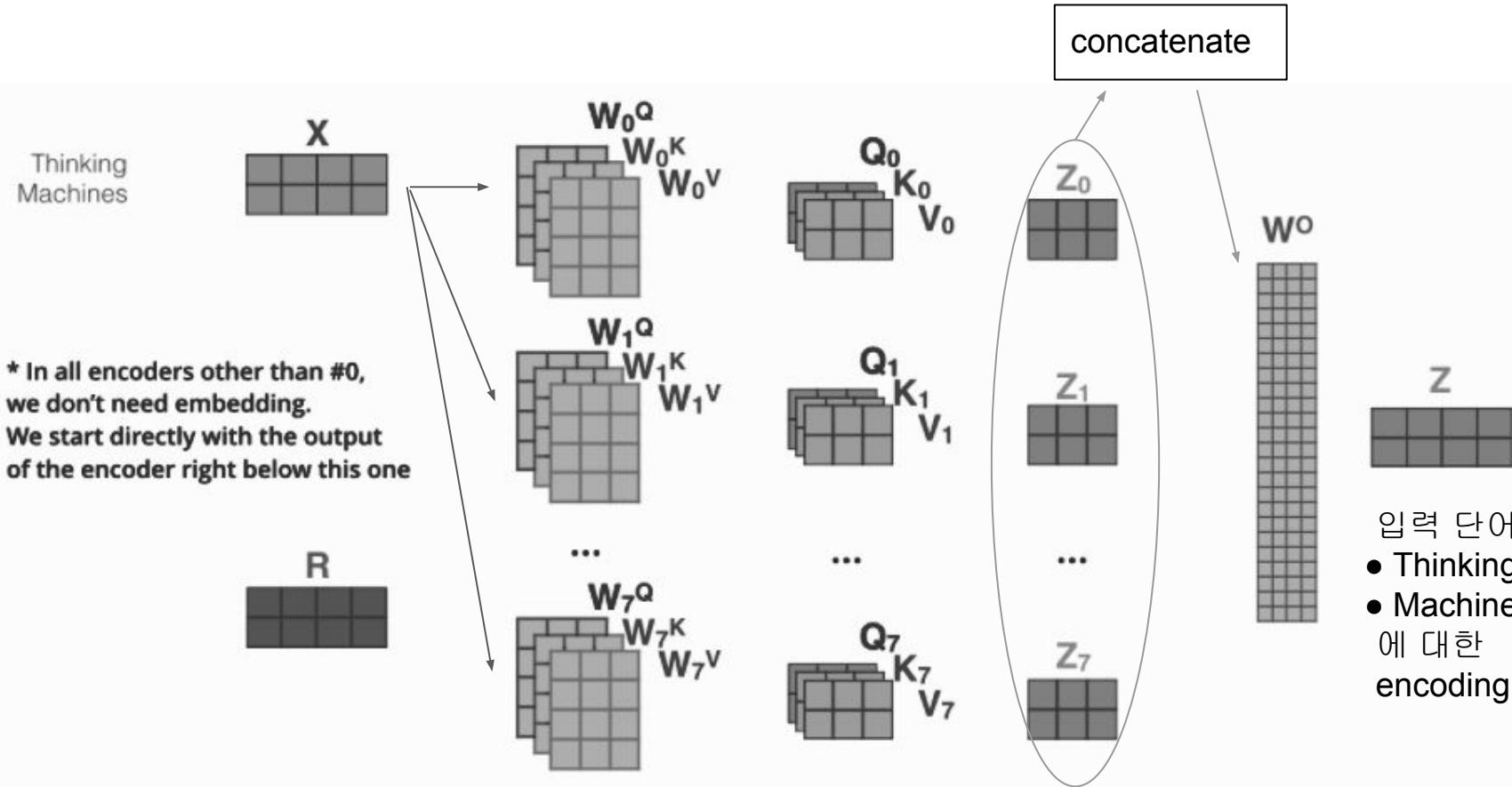
- 논문에서는 Self-attention 메커니즘을 Multi-head로 구현하여 성능을 더욱 개선.
- 자기 자신과의 attention 대신 다른 위치의 단어의 attention을 더 잘 볼 수 있게 됨.
- Self-Attention 계층이 다중으로 여러 representation 공간을 만들 수 있음.



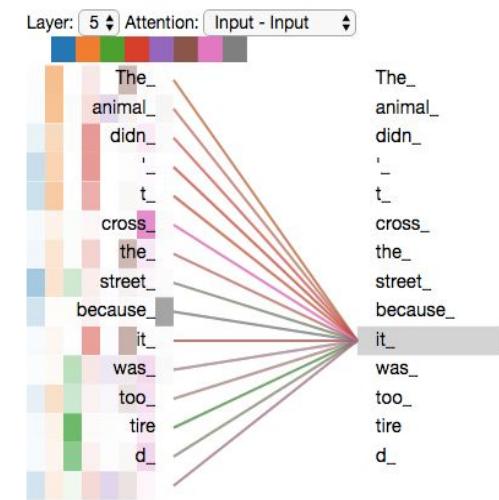
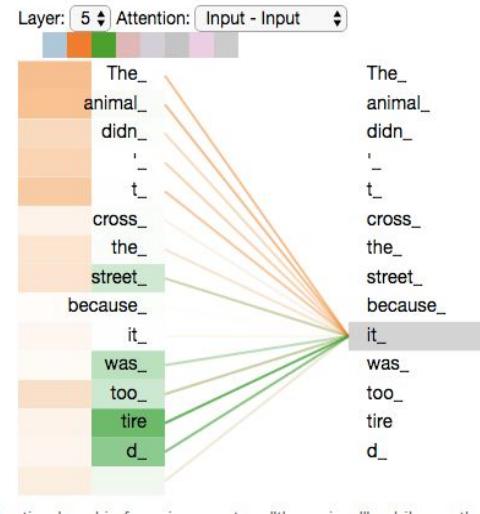
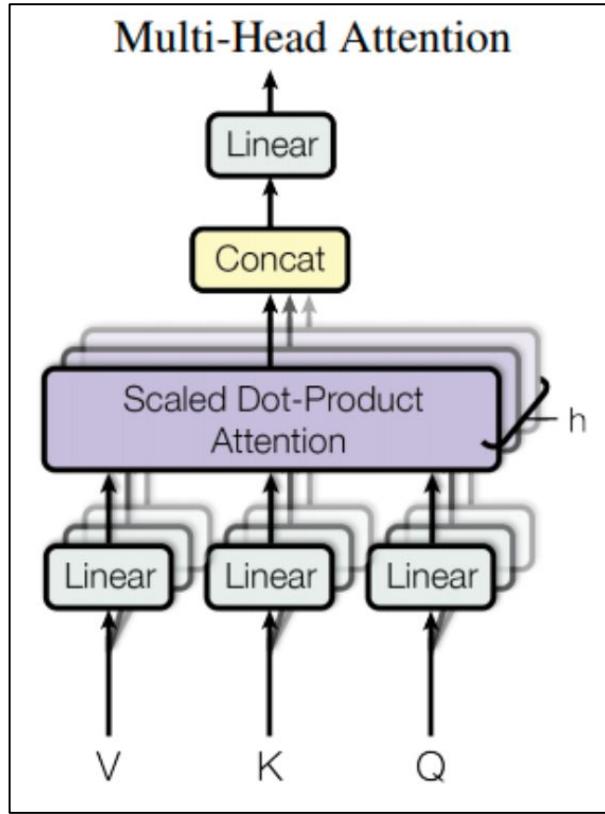
<다중 attention에 의한 출력들>

(1) 출력들은 concatenate  
(2) 행렬곱으로 임베딩 차원에 매핑

# Multi-head self-attention(2)

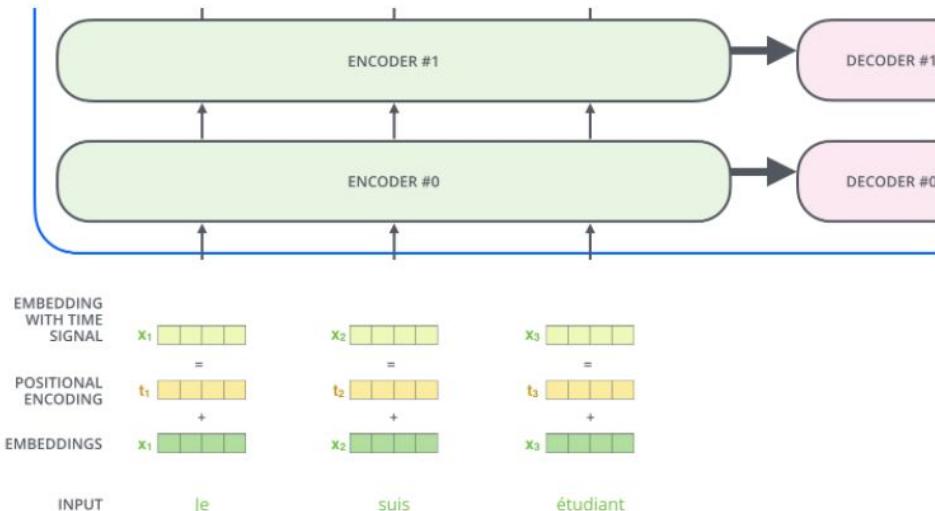


# Multi-head self-attention(3)



# Positional encoding

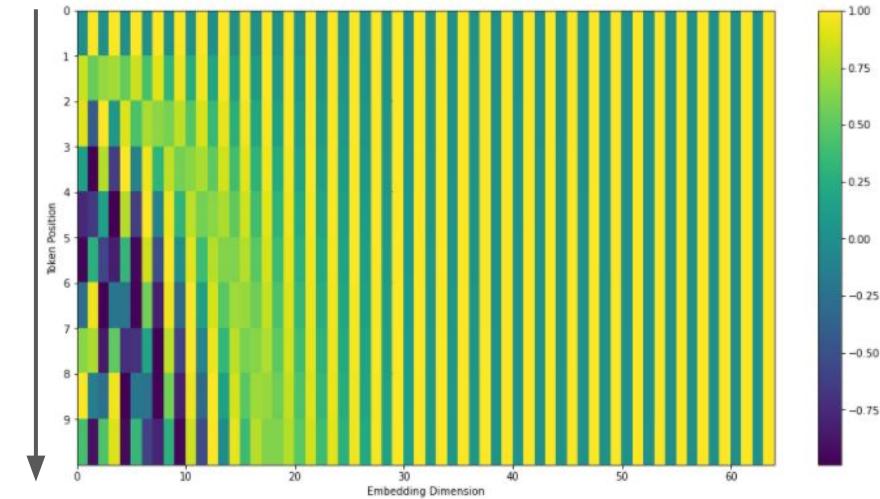
- RNN과 달리 Transformer에는 데이터의 sequence를 고려하지 않음.
- Transformer는 positional encoding을 embedding에 더해 단어의 문장 내 상대적 위치를 encode.



<Encode the positional signals  
into Time signal>

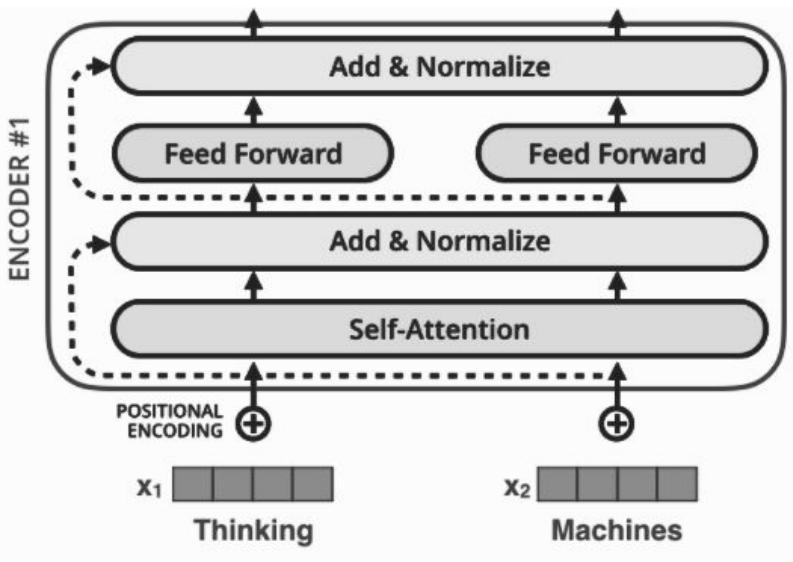
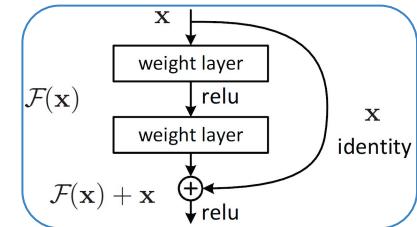
위치에 따른  
인코딩 변화

<Positional signal>

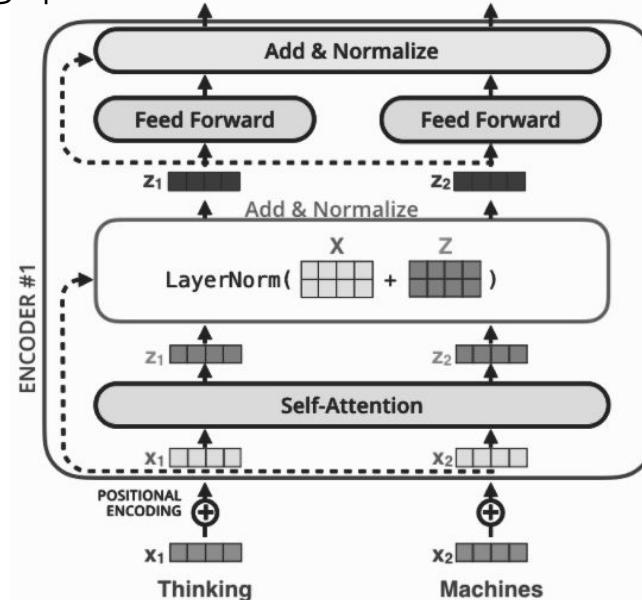


# Residual connections

- 내부 sub layer 사이에 ResNet의 Residual connection 연결
- Encoder가 심층으로 쌓이더라도 경사하강법 학습 용이



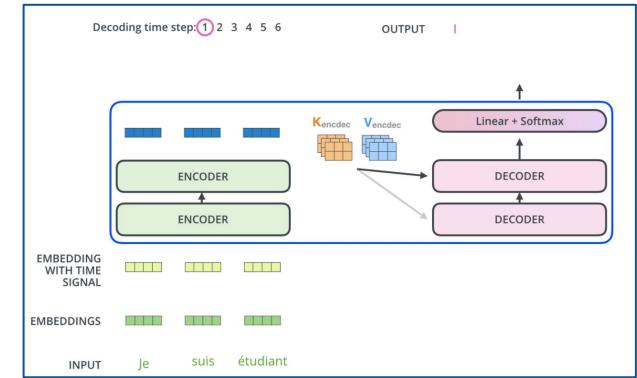
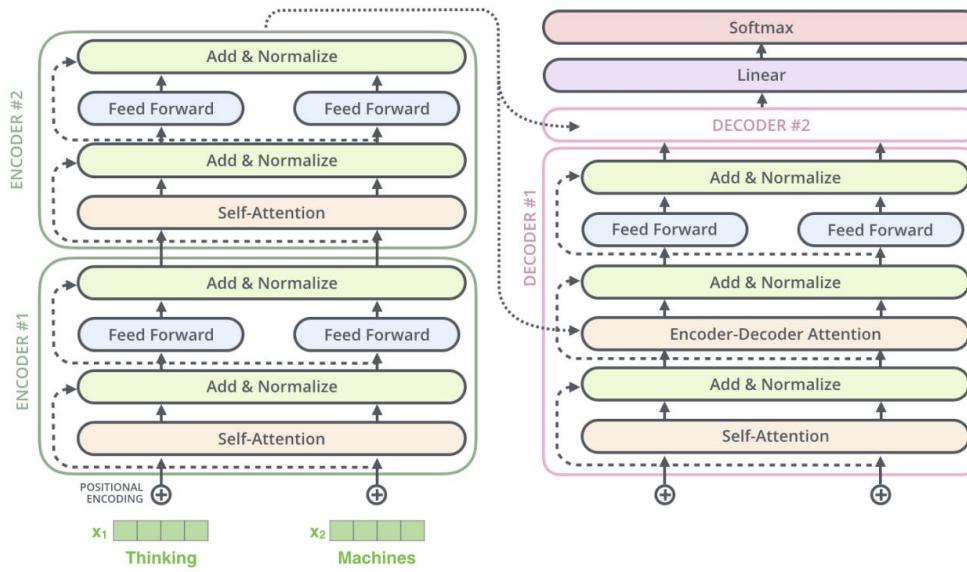
〈Residual connections in Encoder〉



〈Layer Normalization〉

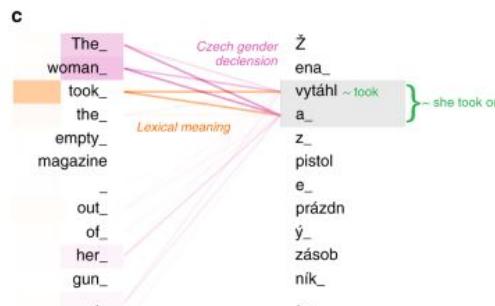
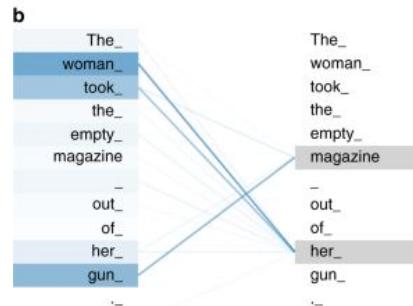
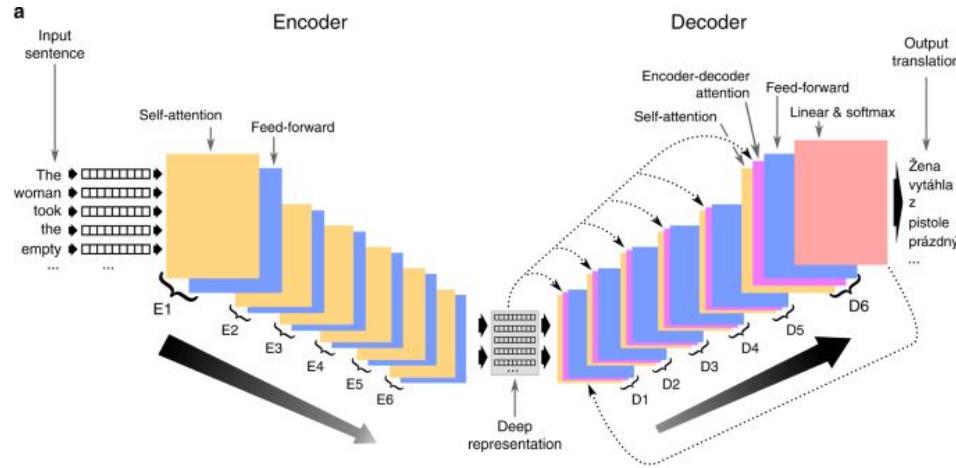
# Transformer : Decoder

- Decoder의 동작은 Encoder와 거의 동일
- Encoder가 입력 시퀀스를 먼저 처리하고, Decoder는 각 스텝마다 출력 시퀀스의 한 element를 출력.
- 차이점은 Encoder-Decoder Attention 계층은 Encoder의 Key와 Value를 사용. <우측 그림>

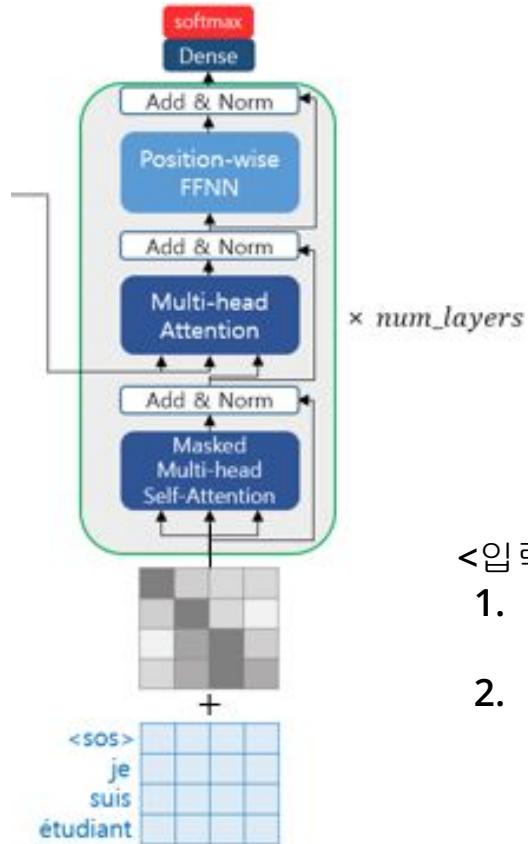


<Encoder-Decoder Attention 계층>  
: Encoder 마지막 레이어의 Key와 Value를 사용.

# Transformer : Decoder



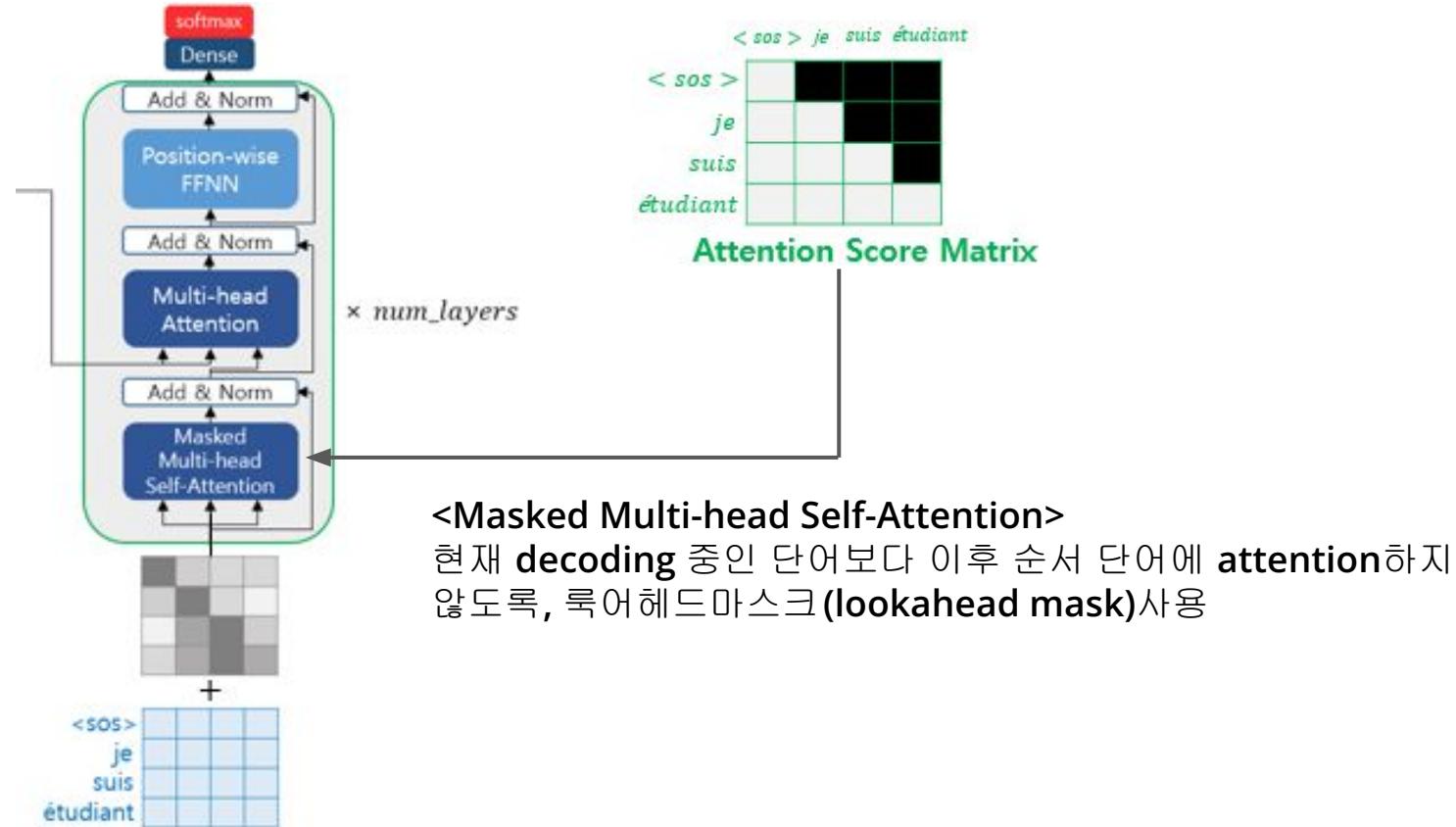
# Transformer : Decoder



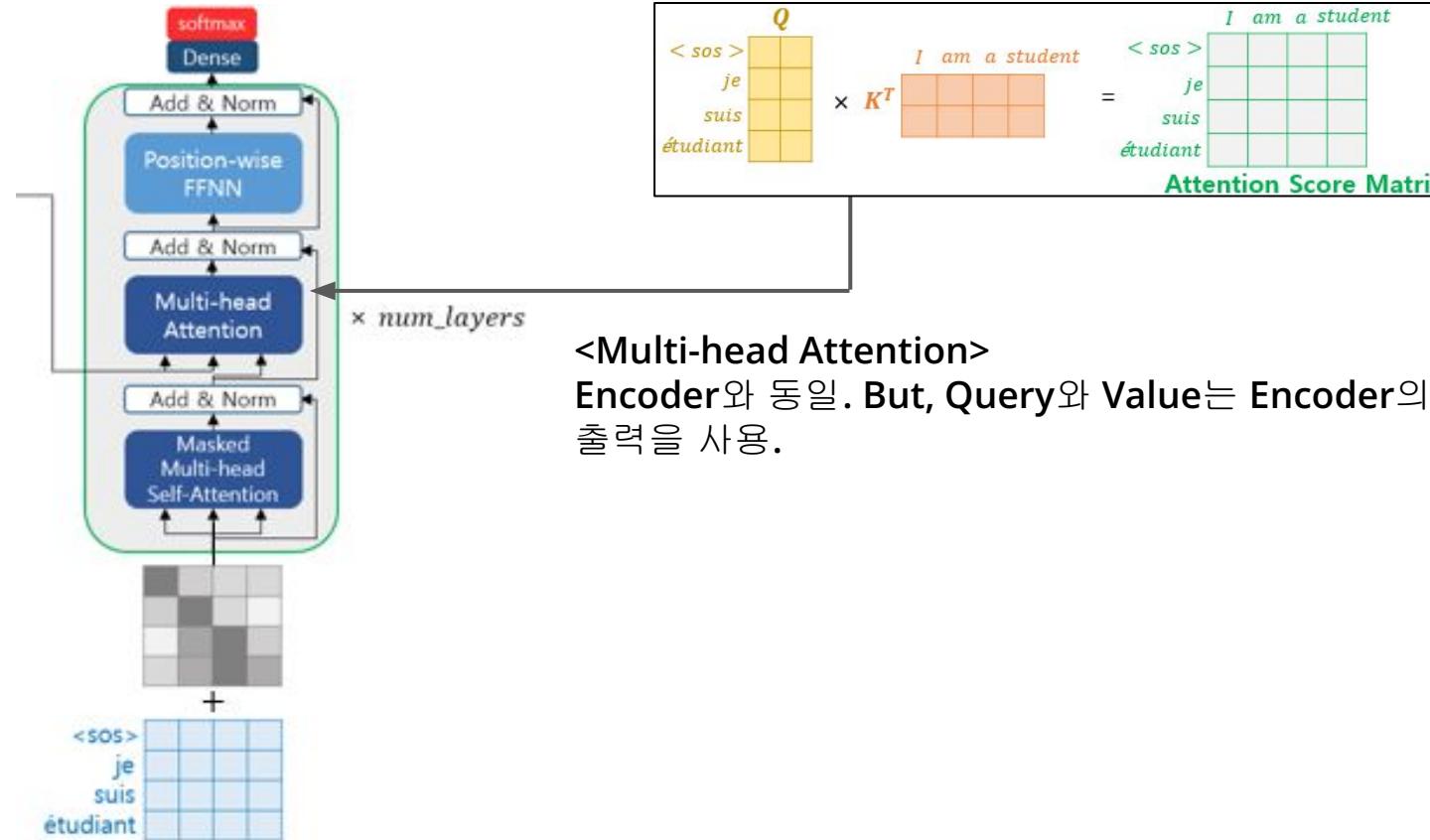
<입력단>

1. Seq2Seq와 같이 Teacher forcing 방식(학습 때는 다음 time step 입력을 정답으로 알려줌)으로 학습.
2. Positional encoding

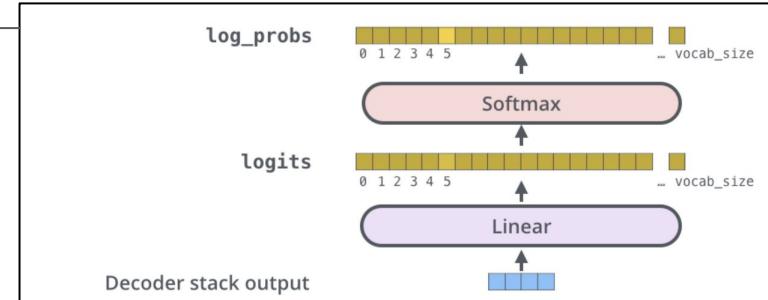
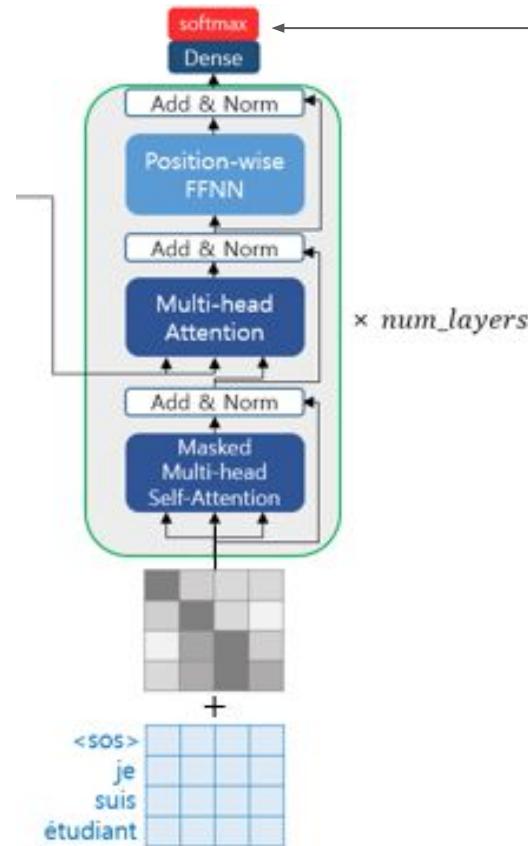
# Transformer : Decoder



# Transformer : Decoder

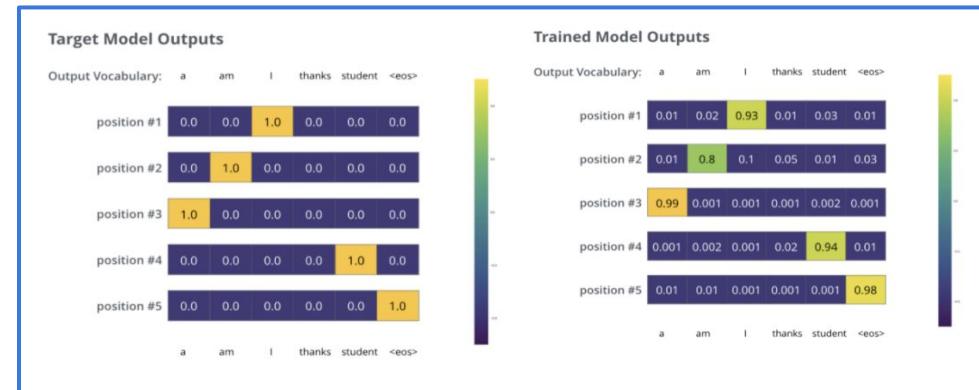


# Transformer : Decoder



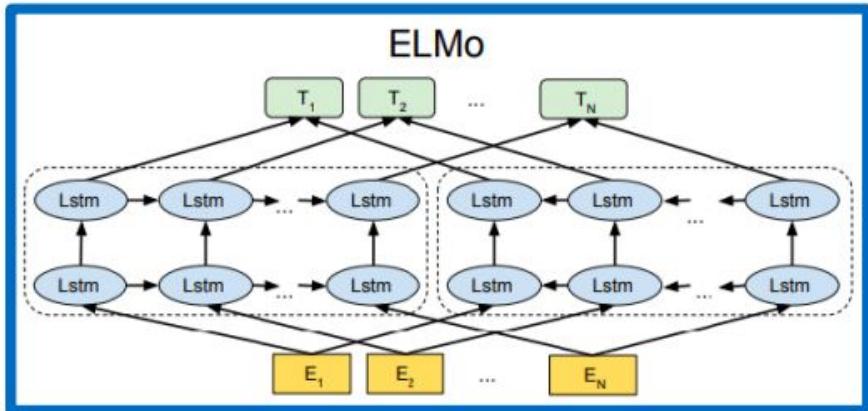
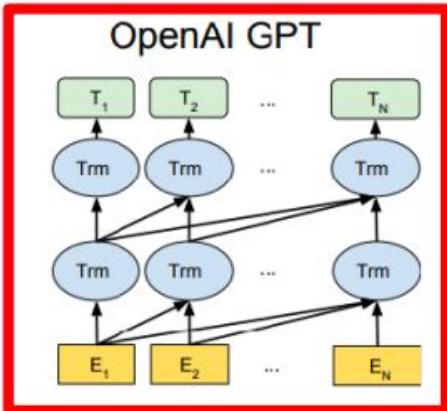
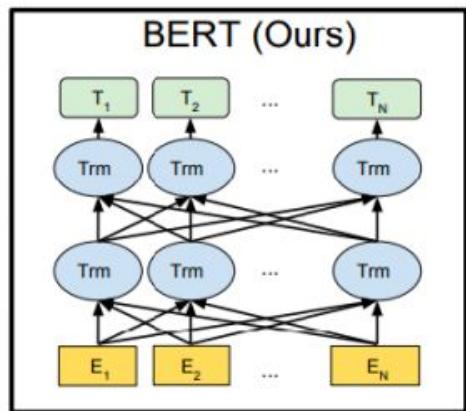
<Softmax after Dense>

각 Step에 대한 cross entropy로 분류 학습.



# Bert & GPT

GPT와 BERT는 모두 자연어 처리 분야에서 사용되는 모델이며 GPT는 문장 생성에 강점을 가지고 있고, BERT는 문장의 의미를 추출하는 데 강점을 지님.

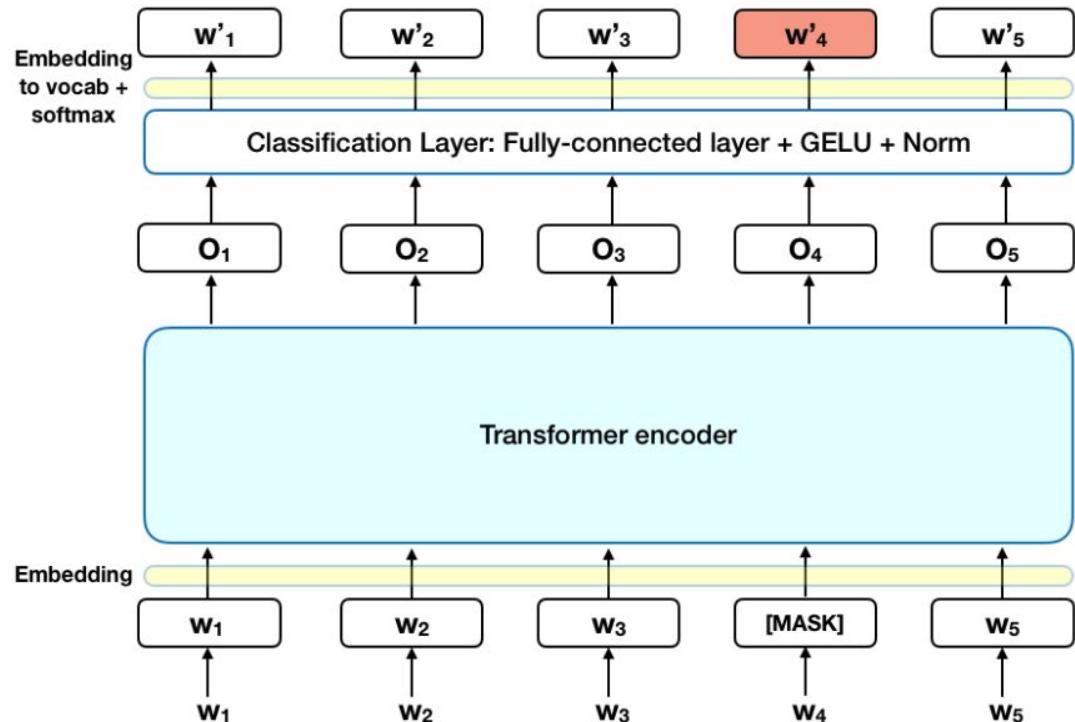


Unidirectional

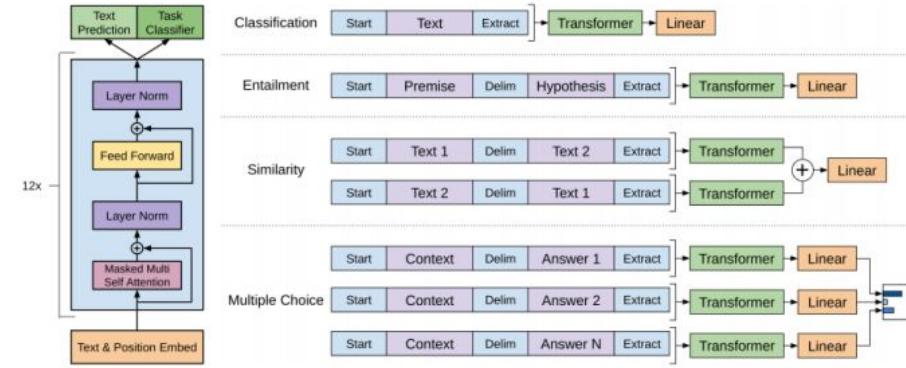
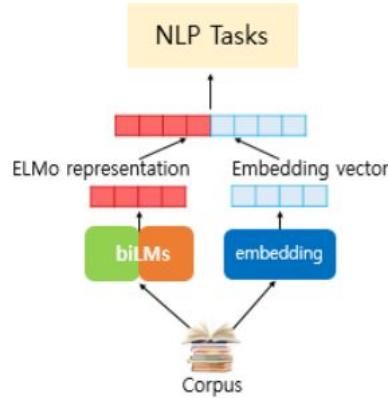
Bi-LSTM

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

- Bert는 12(base)~ 24(large) 개의 transformer 인코더 블록으로 구성
- Bert는 문장의 일부 단어를 스스로 예측해 자연스러운 문장의 구성을 학습하는 방식으로 자연어를 모델링함
- BERT는 양방향 모델 (bidirectional model)로 입력 시퀀스를 양쪽에서 동시에 처리하고, 각 단어를 예측할 때 왼쪽, 오른쪽 문맥 모두를 고려함.



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



## Feature-based

특정 task를 수행하는 network에 pre-trained language representation  
을 추가적인 feature로 제공  
즉, 두개의 network를 붙여서 사용  
e.g. ELMo

## Fine-Tuning

Task-specific한 파라미터를 최소화하고 pre-trained된 파라미터를  
downstream task 학습을 통해 조금만 바꿔주는(fine-tuning) 방식  
e.g. GPT

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

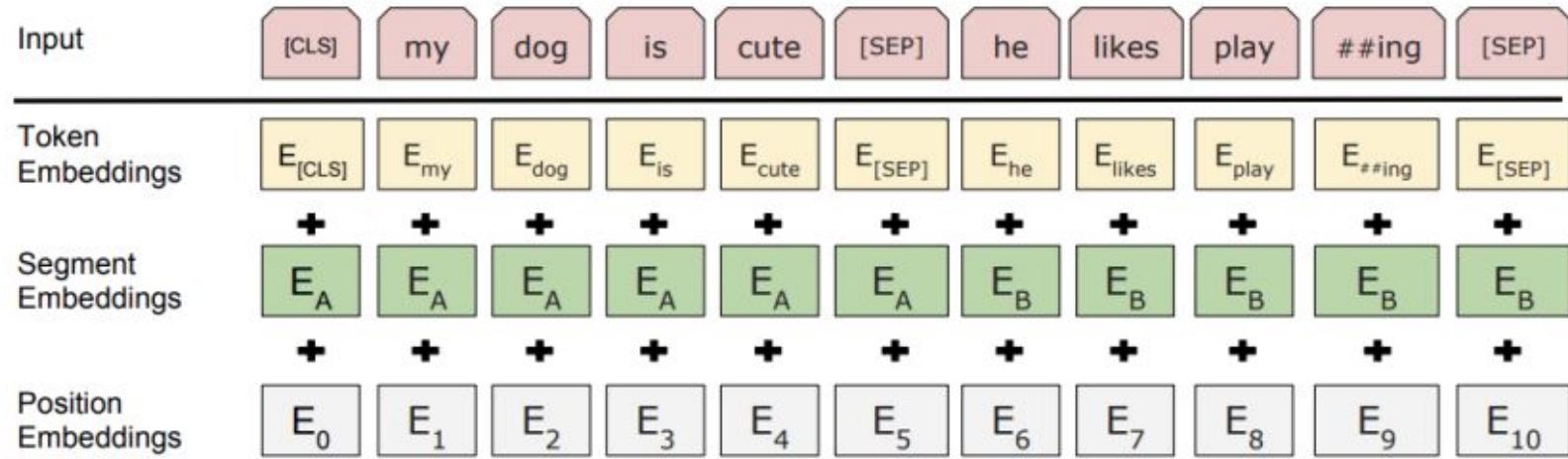


Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

- Tokenizer : WordPiece Tokenizer
- 첫 번째 토큰: [CLS] → self-supervised learning을 거쳐 입력 문장의 embedding으로 문장 전체의 내재적 구조를 이해할 수 있는 토큰
- [SEP]: 두 개의 문장을 입력할 때 그들을 구분하기 위한 token

# Bert : self-supervised learning

- Masked language Model(MLM)
    - 일종의 Noise AutoEncoder 방식의 학습으로 해석 가능.
    - 15%의 일부 token을 masking해 masked token을 예측
    - 15% 중,
      - 80%는 [MASK] e.g my dog is hairy => my dog is [MASK]
      - 10%의 경우 : random word e.g my dog is hairy => my dog is apple
      - 10%의 경우 : 원래의 단어
    - 단어가 문장 안에서 가지는 맥락을 파악해 단어의 중의성 문제를 해결
      - e.g 밤을 먹다. vs 밤이 되었다.
      - 같은 단어, 같은 위치에 있더라도 서로 다른 embedding 벡터를 가지게 됨.

# Bert : self-supervised learning

- Next Sentence Prediction (NSP)

- 두 문장이 주어졌을 때 이어지는 문장인지 여부를 예측
- 50%: sentence A, B가 실제 next sentence
- 50%: sentence A, B가 corpus에서 random으로 뽑힌(관계가 없는 두 문장)

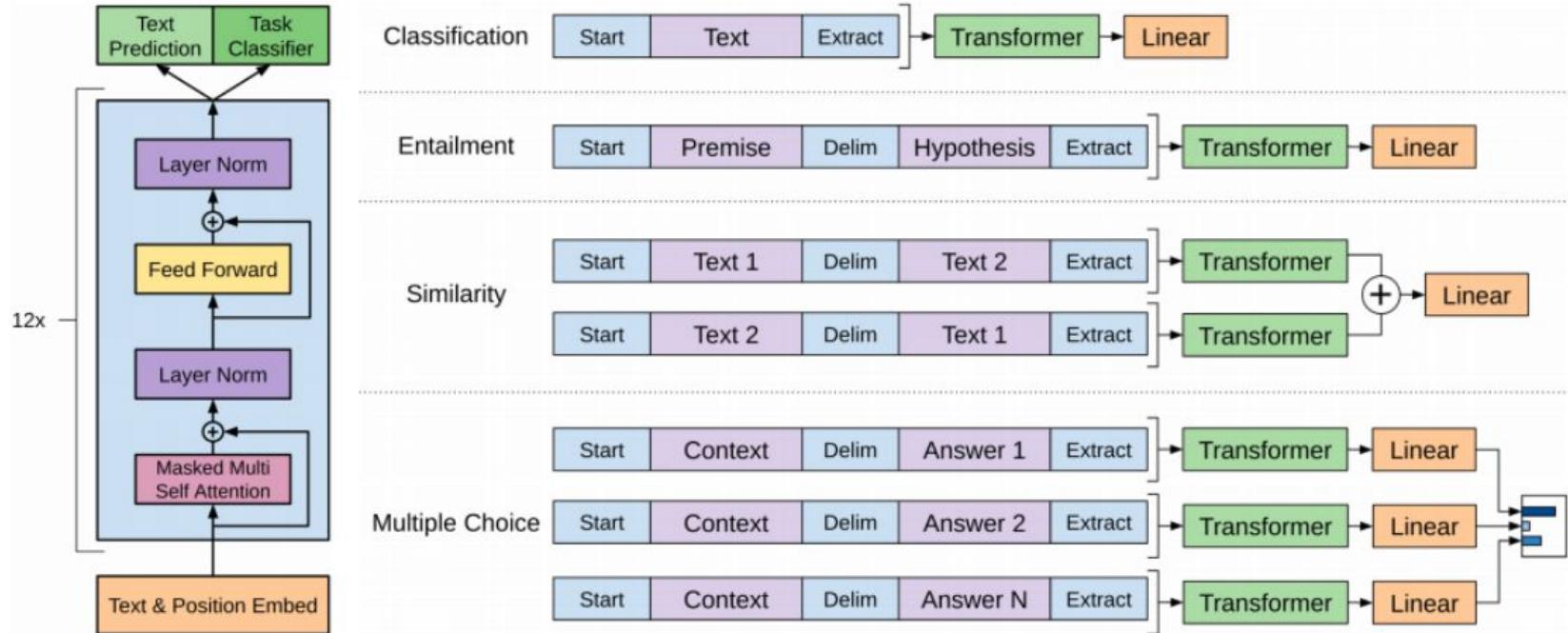
```
Input = [CLS] The man want to [MASK] store [SEP] he bought a gallon  
        [MASK] milk [SEP]
```

```
Label = IsNext
```

```
Input = [CLS] The man want to [MASK] store [SEP] penguin [MASK]  
        are flight less birds [SEP]
```

```
Label = NotNext
```

# GPT : Improving Language Understanding by Generative Pre-Training



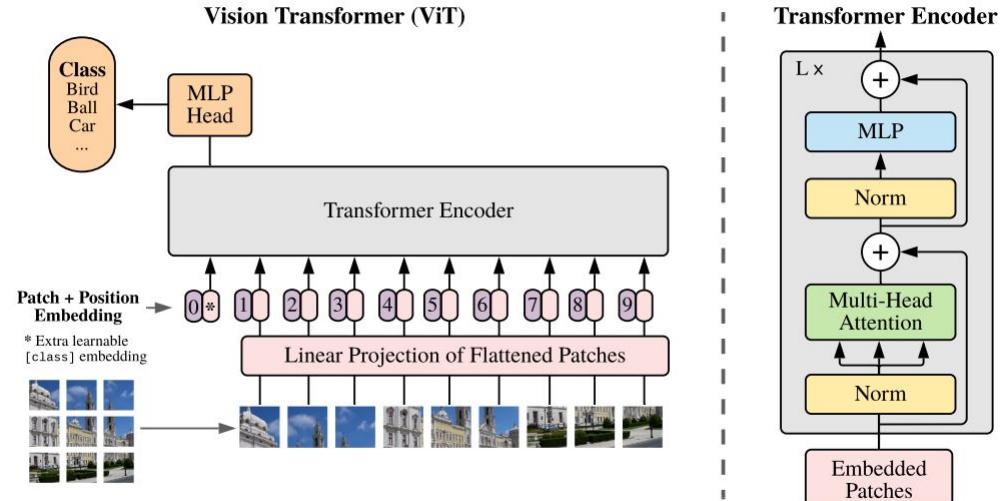
# GPT : Improving Language Understanding by Generative Pre-Training

- GPT는 Transformer의 Decoder로 구성되어 각 입력 단어에 대한 출력이 다음 단의 예측 확률에 영향을 줌. 따라서 문장생성에 특화되어 있어 Text-to-Text 형태의 문제에서 탁월한 성능을 나타냄.
- GPT의 학습은 크게 두 단계로 구성.
  - 사전 학습(**pre-training**) 단계
    - 대규모의 텍스트 데이터에 노출시켜 일반적인 언어 모델을 학습. GPT 모델은 텍스트 데이터의 패턴과 구조를 학습하여, 문맥을 이해하고 다음 단어를 예측하는 능력을 갖춤.
  - 미세 조정(**fine-tuning**) 단계
    - 사전 학습된 GPT 모델을 특정 자연어 처리 작업에 맞게 fine-tuning하여 해당 작업을 수행하는 데 사용됨. 이 과정에서는 적은 양의 레이블된 데이터셋으로 GPT 모델을 학습시켜 해당 작업에 최적화된 모델을 생성.
- GPT는 모델의 크기의 증가가 downstream 성능을 증가시키므로 GPT-1이후 GPT-4까지 마이너한 레이어 수정을 제외하면 모델 규모의 증가를 뜻함.
- 실제 GPT-3 이후에서는 few-shot 또는 zero-shot에도 좋은 성능을 나타냄.

# Vision Transformer(ViT)

Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020).

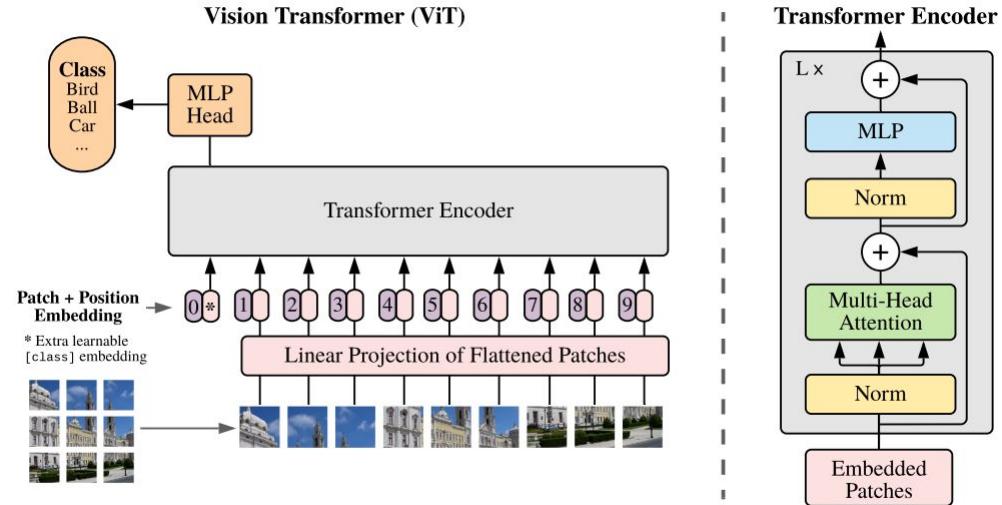
- Vision Transformer는 이미지 처리를 위해 Bert 구조를 사용하되 sequence 토큰을 입력 이미지의 patch로 대체
- Vision Transformer의 장점은 확장성이 좋으며, 대규모 스케일 학습에서 CNN 보다 한계 성능이 더 우월함.
- Inductive bias의 부족으로 학습을 위해 CNN 보다 더 많은 데이터가 필요하다는 단점 있음.



# Vision Transformer(ViT)

Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020).

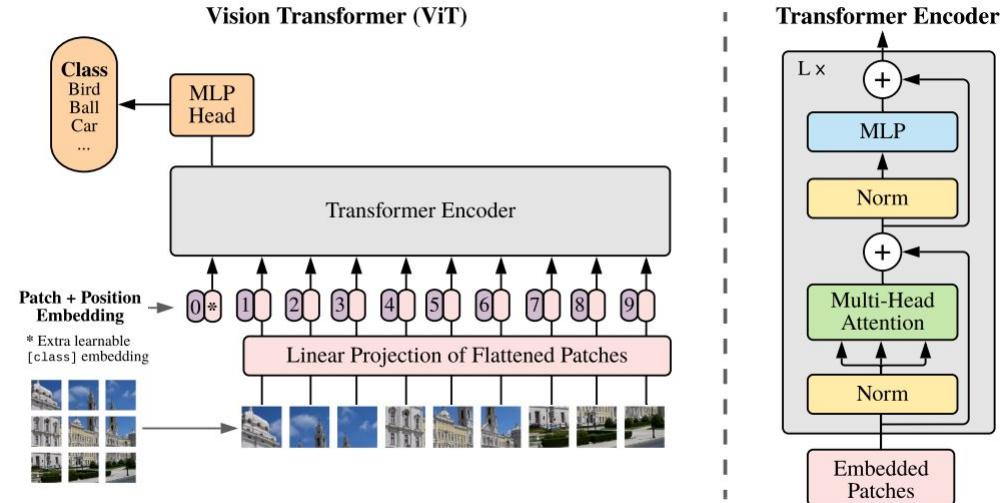
- 입력 사진을 **16x16 패치**로 분할 후 평탄화(**flatten**)해 각 패치를 단어처럼 다룸.  $(16,16,3) \Rightarrow (768)$
- Flatten된 패치는 **linear embedding** 수행.  $(768) \Rightarrow (128)$
- 언어모델과 같이 각 patch signal에 **Positional embedding**을 더함.  $(128) \Rightarrow (128)$



# Vision Transformer(ViT)

Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020).

- 논문에서는 **303M**개 이미지와 **18K** 클래스가 포함된 **Google**의 비공개 데이터 **JFT-300M**을 사용.
- ImageNet으로 pre-training을 했을 때는 그리 좋은 성능을 보이지 못했음. Transformer는 CNN과 달리 **Inductive bias**가 없기에 더 많은 데이터가 필요



# Self-supervision on vision tasks

- Contrastive learning
- SimCLR v1, v2
- MAE

# Self-supervised learning for vision tasks

- 최근 비지도학습(**unsupervised learning**)보다 자기주도학습(**self-supervised learning**)이라는 용어가 더 잘 받아들여지고 있음. 자기주도학습은 정답 생성이 가능한 문제를 스스로 풀어보는 과정에서 자연스러운 데이터 분포가 무엇인지 학습.
- 가장 간단한 방식은 모델 학습을 위한 새로운 형태의 문제인 **pretext task**를 학습
  - 이미지를 회전시키고 회전 방향과 각도를 맞추도록 학습
  - 이미지를 잘라 **zigsaw** 퍼즐을 만들고 모델이 퍼즐을 풀도록 학습
- 그러나 **pretext task**를 통해 학습하는 방식은 **pretext task**를 잘 풀게끔 학습될 뿐 이미지의 일반적인 시각 특징을 포착하는데 실패해왔음.

# Self-supervised learning for vision tasks

- 최근 **self-supervised learning** 방식으로 **vision task**를 학습하기 위한 가장 유망한 방법 중 하나는 **Contrastive learning**이다.
- 예시처럼(**cat vs. dog**) 우리가 비슷한 것과 다른 것을 비교할 수 있다면 고수준 특징을 알고 있다는 의미함.



=

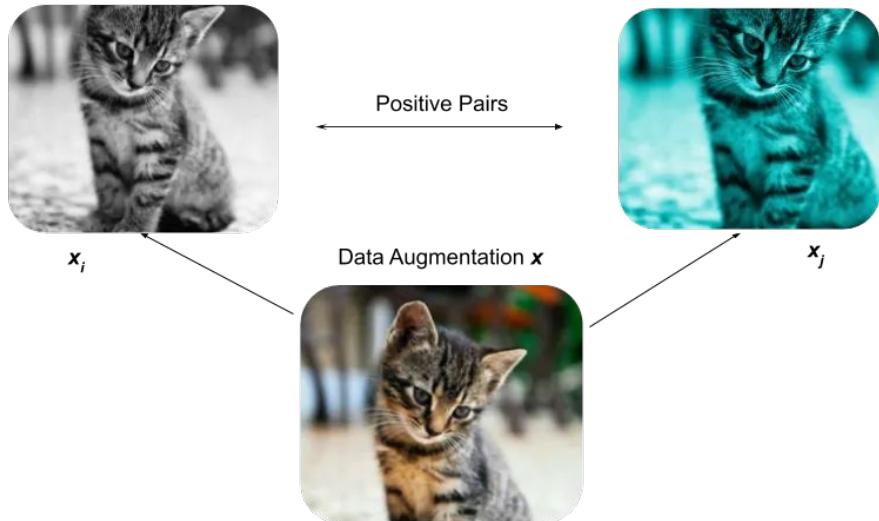


≠



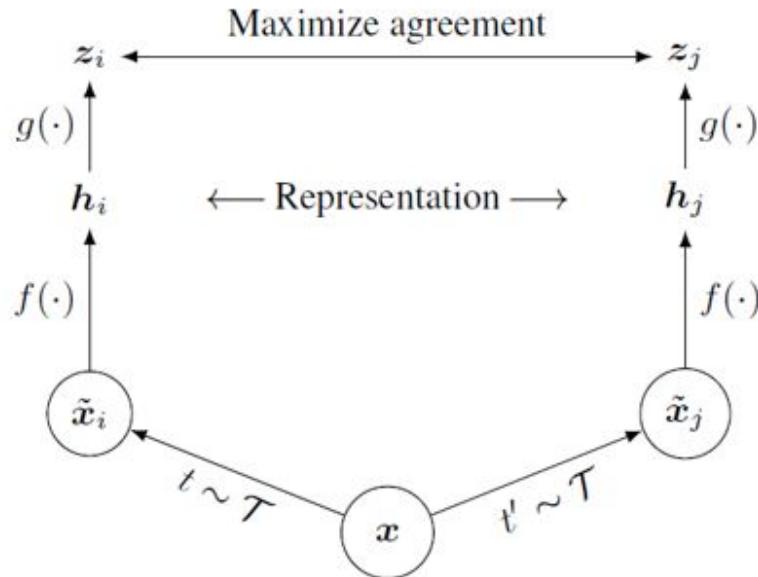
# Self-supervised learning for vision tasks

- 핵심적인 아이디어는 동일한 사진에서 나온 데이터 증강의 결과들을 **positive pair**로, 다른 사진에서 나온 결과들은 **negative pair**로 간주하는 것.



# Self-supervised learning for vision tasks

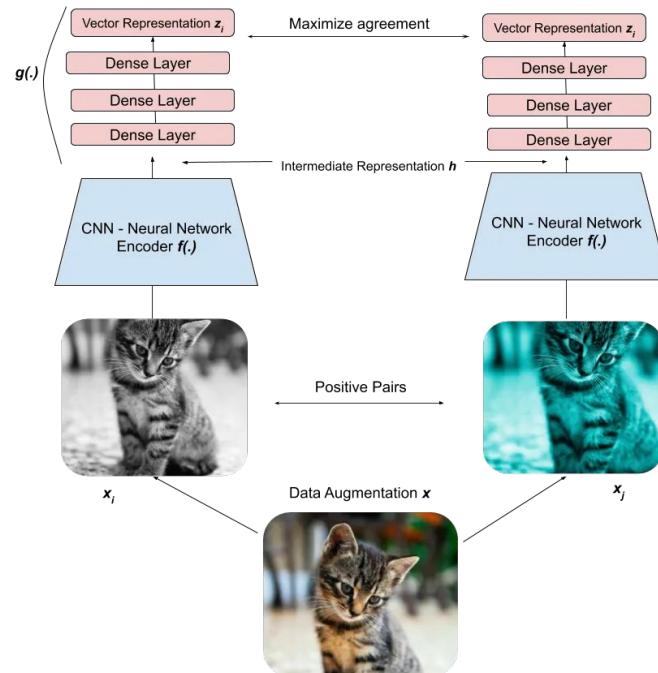
- 즉 **Contrastive learning**은 모델에게 데이터들이 비슷하다, 다르다를 학습시켜 데이터의 레이블 없이도 데이터 특징을 모델링할 수 있는 머신러닝 기법임.
- 때문에 **Contrastive learning**은 레이블 없이도 대규모 데이터의 특징을 학습할 수 있으므로 많은 학술적 주목을 받고 있음.



# SimCLR

Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." International conference on machine learning. PMLR, 2020.

- SimCLR은 Google Research에서 발표한 **contrastive learning**을 위한 학습 구조
- 하나의 이미지에서 무작위적으로 증강한 두 개의 이미지는 서로 **positive pair**로 정의



# *Image augmentation in SimCLR*



(a) Original



(b) Crop and resize



(c) Crop, resize (and flip)



(d) Color distort. (drop)



(e) Color distort. (jitter)



(f) Rotate  $\{90^\circ, 180^\circ, 270^\circ\}$



(g) Cutout



(h) Gaussian noise



(i) Gaussian blur

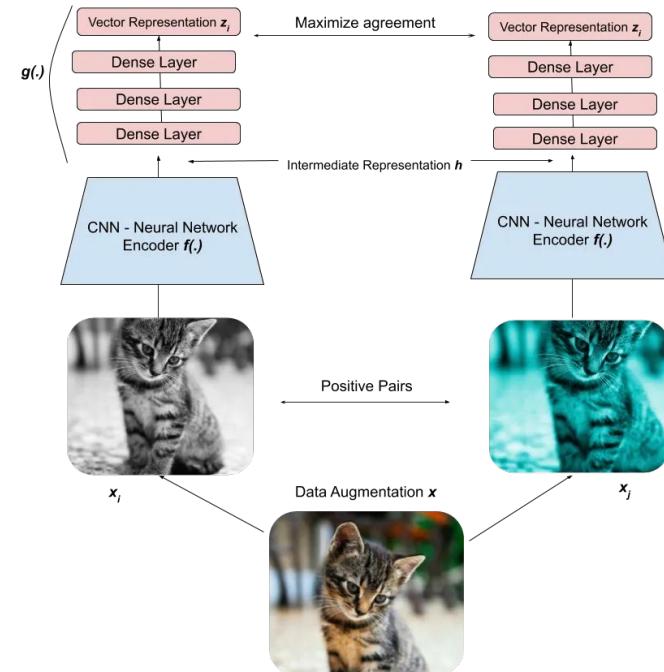


(j) Sobel filtering

# SimCLR

Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." International conference on machine learning. PMLR, 2020.

- 만약  $N$ 개의 batch size를 사용할 때 각 샘플 별로 1개의 positive pair와  $2N-2$ 쌍의 negative pair를 구성할 수 있음.
- SimCLR은 positive pair 사이의 유사도 함수를 키우고 negative pair 사이의 유사도 함수를 최소화하도록 학습.
- Contrastive learning에서 충분히 많은 negative pair가 중요함. 논문에서는 4096개의 batch size를 사용해서 학습했음.



# Loss function in SimCLR

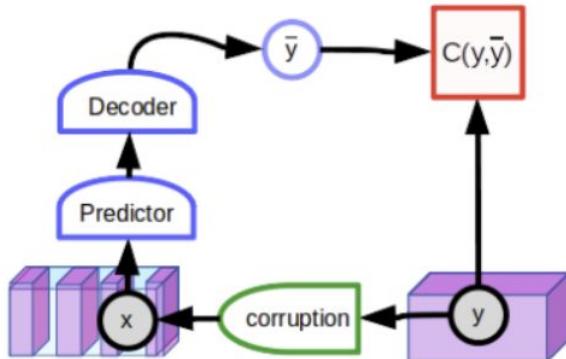
$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$

Negative pairs

Concatenated views	Image 1 view 1	Image 2 view 1	Image 1 view 2	Image 2 view 2
Image 1 view 1		Negative pair	Similarity image 1	Negative pair
Image 2 view 1	Negative pair		Negative pair	Similarity image 2
Image 1 view 2	Similarity image 1	Negative pair		Negative pair
Image 2 view 2	Negative pair	Similarity image 2	Negative pair	

# Self-supervised learning for vision tasks

- 또 다른 유망한 접근방법은 데이터에서 드러난 패트에서 숨겨진 패트를 추론하도록 학습하는 것.
  - 예를 들어 자연어처리에서는 전체 문장 중 일부 단어를 숨기고, 나머지 단어로 숨겨진 단어들을 추론해 복원하도록 학습해 임팩트 있는 성과를 얻음(**BERT**, **RoBERTa**, **XML-R**)



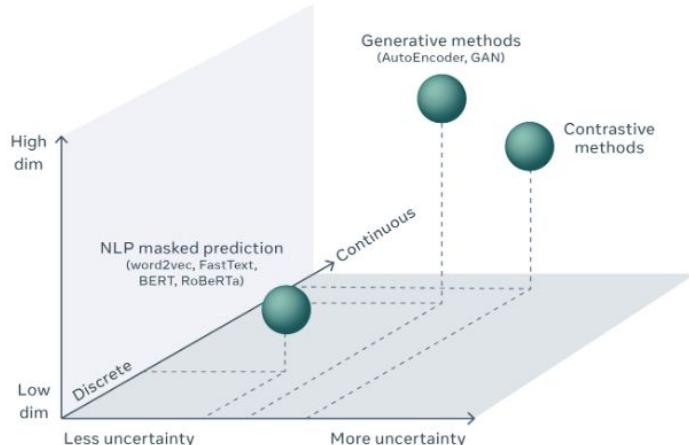
This is a [...] of text extracted  
[...] a large set of [...] articles

This is a piece of text extracted  
from a large set of news articles

Decoder는 주변 단어를 바탕으로 빈칸의 단어(단어사전의 60k~100k 중 1) 예측. 방대한 문장이 주어진다면 유의어 관계를 이해할 수 있을 정도의 확률이 주어짐.

# Self-supervised learning for vision tasks

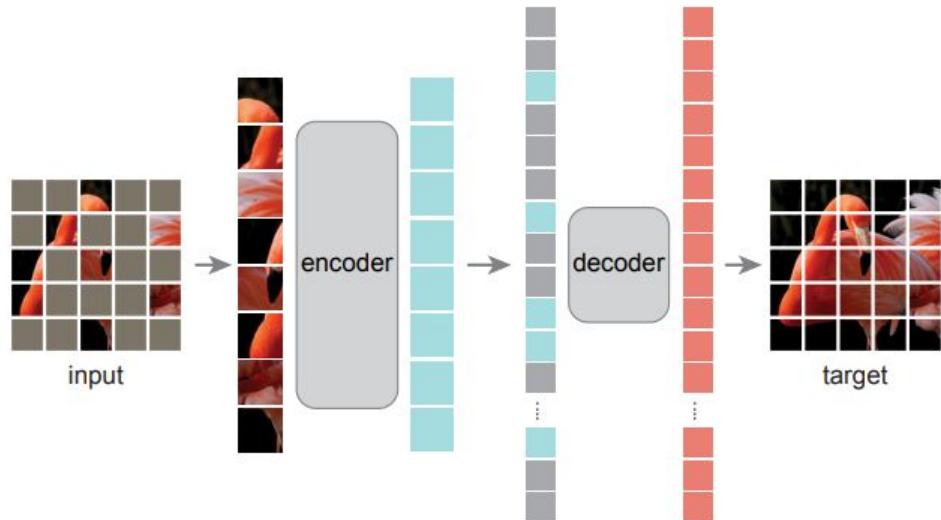
- 반면 컴퓨터비전 분야에서는 같은 수준의 진보를 가져오지 못 했는데 주된 이유는 단어들보다 이미지들에서 예측의 불확실성이 높기 때문
  - 자연어처리에서 손실된 단어의 예측은 거대한 단어사전 중 해당 빈칸에 들어갈 수 있는 단어에 대한 예측이므로 해당 문제에 대한 불확실성은 단어사전 내 모든 가능한 출력 상의 확률 분포가 됨.
  - 비디오에서 손실된 프레임들의 예측, 이미지에서 손실된 패치들의 예측, 음성신호에서 손실된 세그먼트의 예측은 고차원 연속값 객체에 대한 예측으로, 가능한 모든 예측들을 특정하는 것이 불가능



# MAE(Masked AutoEncoder)

He, Kaiming, et al. "Masked autoencoders are scalable vision learners." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.

- 2022년 Facebook AI Research 팀은 masked language modeling 기법을 vision task에 적용 성공
- 언어 모델링과 마찬가지로 데이터(이 경우 사진)의 일정 부분을 제거하고, 모델이 제거된 부분을 예측하도록 학습함.

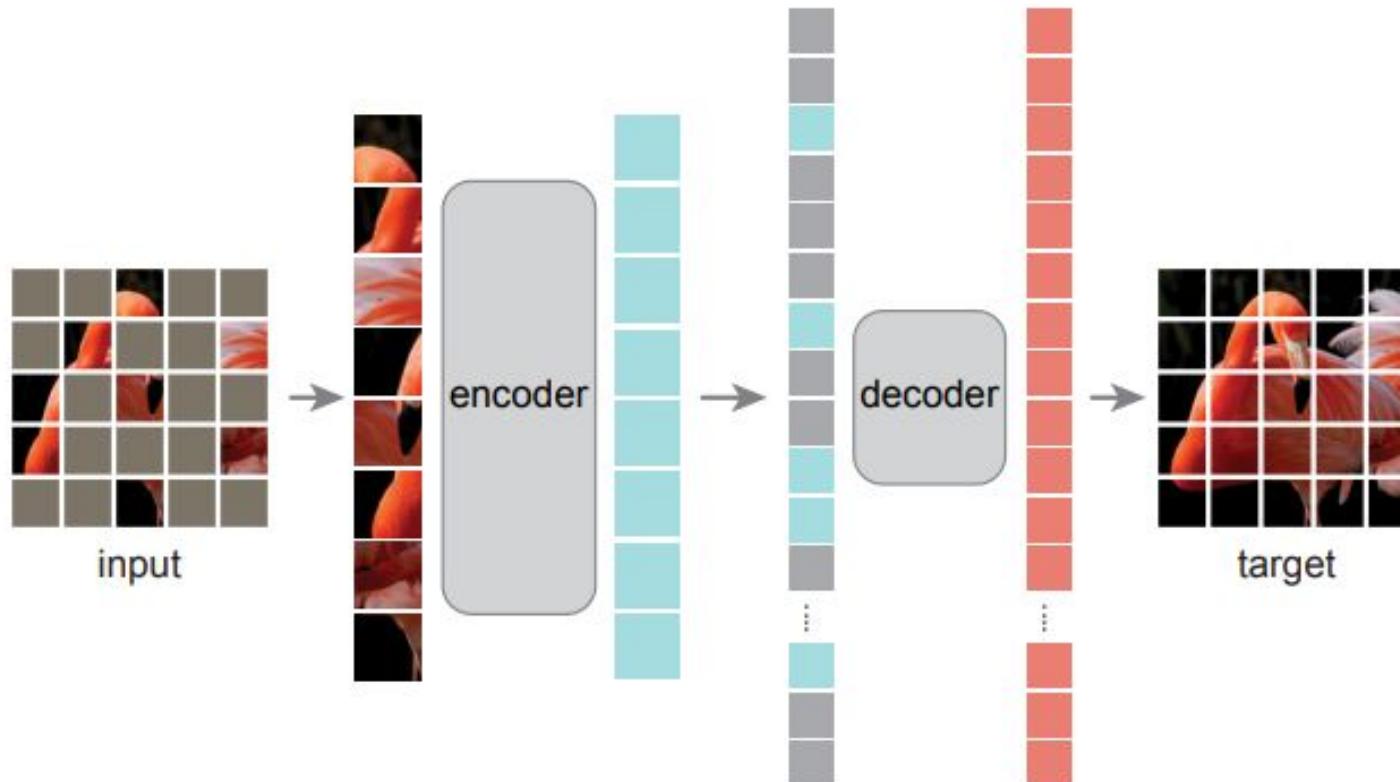


# MAE(Masked AutoEncoder)

He, Kaiming, et al. "Masked autoencoders are scalable vision learners." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.

- 저자들은 비전 모델에 masked modeling이 적용되지 않았던 이유를 다음과 같이 설명함.
  - 지금까지 시각모델로 CNN이 지배적이었기 때문에 구조적 차이가 있었음. 그러나 이제 ViT가 개발됐기 때문에 해결되었음.
  - 자연어와 시각은 정보 밀도가 굉장히 다르다. 자연어는 인간에 의해 만들어졌기 때문에 훨씬 의미론적이고 정보 밀도가 높다.
    - 제거된 단어를 예측하는 행위는 의미론적 정보를 학습하도록 유도하지만, 일부 제거된 시각정보를 복원하는 행위는 인지적 정보인 주변 영상 패치들에 더 의존할 수 있다.
- 이를 극복하기 위해 전체 넓이의 75%라는 매우 높은 비율로 영상의 픽셀을 제거하였음. 이것은 놀랍게도 다음 두 가지 이점이 있음.
  - 영상의 매우 많은 정보가 제거됐기 때문에 모델은 사진에서 의미론적 정보를 해석하도록 강제
  - 동일한 이유로 학습 시 연산량은 매우 감소시킴(x3).

# *MAE architecture*



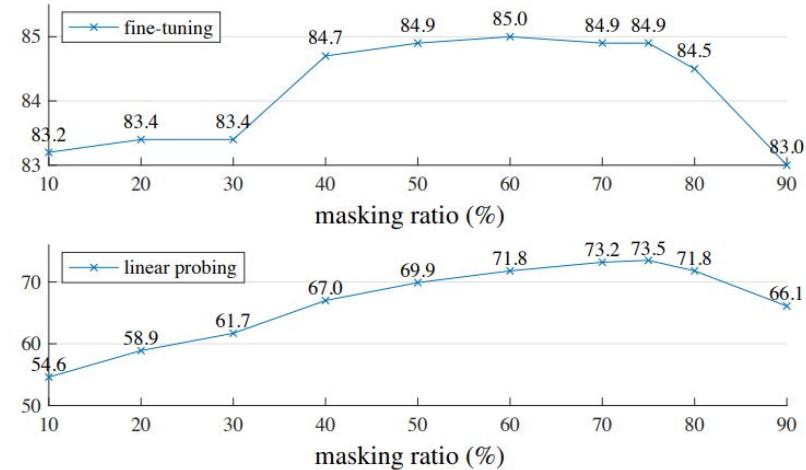
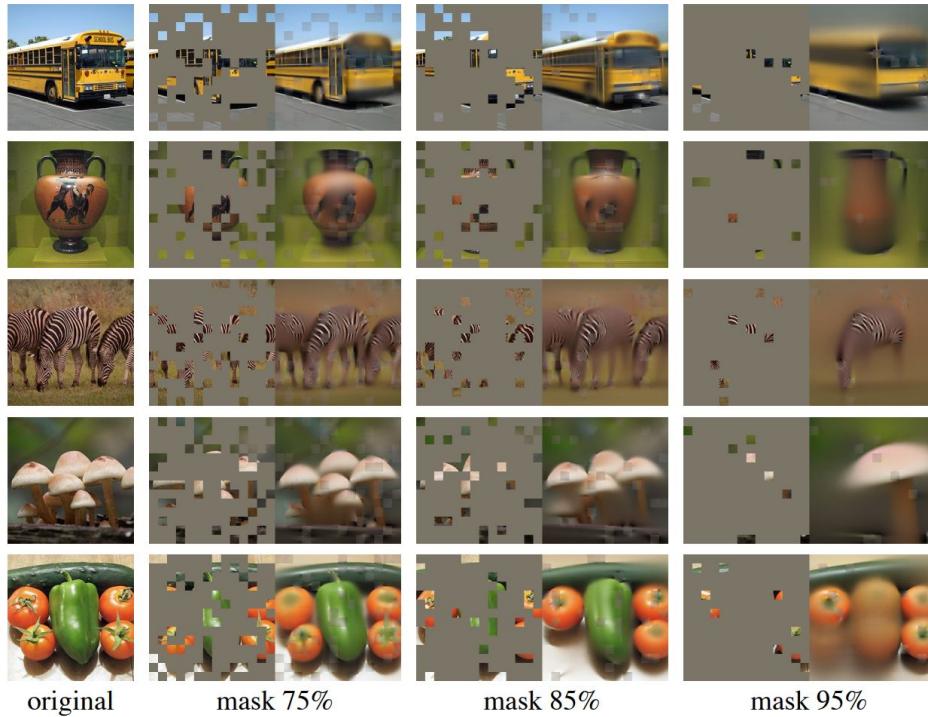


Figure 5. **Masking ratio.** A high masking ratio (75%) works well for both fine-tuning (top) and linear probing (bottom). The y-axes are ImageNet-1K validation accuracy (%) in all plots in this paper.

# Vision-language model pretraining

- CLIP
- Prompt learning
- Using Hierarchical label sets

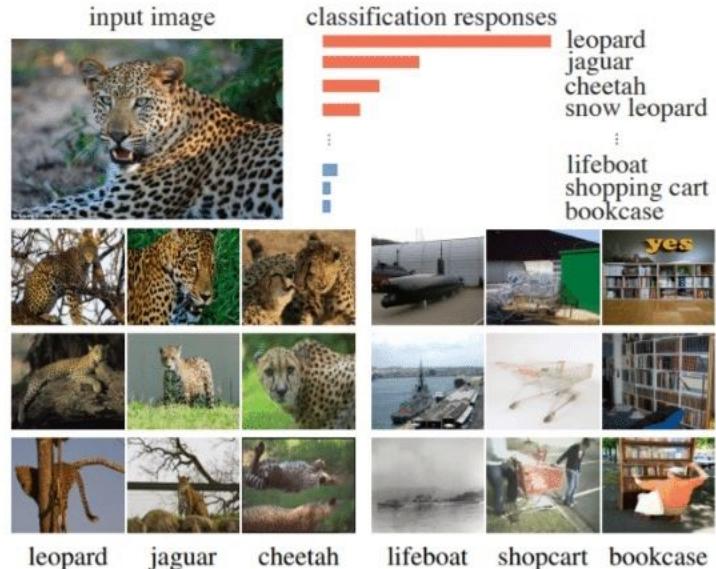
# CLIP : Contrastive Language-Image Pretraining

What is the motivation behind CLIP? 왜 비전모델은 언어모델에 비해 재사용성이 낮은가?

- GPT-3와 같은 최신 **text-to-text** 언어모델들은 특정 작업이나 데이터셋에 특화되지 않고 다양한 다운스트림 작업들에서 잘 작동함.
  - Autoregressive modeling, masked language modeling과 같은 **Task-agnostic learning** 사용
- 대규모 웹 텍스트를 학습한 **Task-agnostic pre-trained model**을 구축하는 전략이 작업자가 생성한 높은 품질의 **NLP** 데이터셋을 구축하는 것보다 유리할 수 있음을 시사.
  - 새로운 작업에 대한 지도학습 데이터가 대규모로 필요하지 않음.
  - Zero-shot transfer, Transfer learning
- 반면 비전모델은 언어모델에 비해 일반화(generality)와 사용성(reusability)이 낮았음.

# CLIP : Contrastive Language-Image Pretraining

- 전형적인 비전데이터 ImageNet의 경우
  - 구축을 위해 노동집약적, 고비용의 문제
    - 22,000 객체의 1400만 여장의 사진을 레이블링하기 위해 25,000명의 작업자 참여
    - 하위데이터셋 ImageNet-1k는 1000 카테고리의 1,281,167장 학습이미지로 구성.
  - 제한된 카테고리의 문제
    - 사전정의된 1-of-N 카테고리 레이블 학습
    - 좁은 시각적 개념만을 학습



# 자연어 지도학습(Natural language supervision) (1/2)

더 다양한 시각개념을 사용하기 위해 인간의 언어를 활용할 수 없을까?

- 자연어(캡션, 태그, 키워드, 제목 등)를 학습 레이블로 사용하는 자연어 지도학습은 시각표현을 학습하는 새로운 학습방법이 될 수 있음.
- 자연어 지도학습 기반 시각모델 학습에 대한 연구는 20여년 간 존재. 그러나 여러 제약으로 성능 한계 존재
  - 특히 **topic model** 또는 **n-gram**과 같은 기존 언어모델이 자연어의 복잡한 맥락을 다루는데 불충분(McCann et al., 2017)

Image	Given Tags	Generated Tags
	pentax, k10d, kangarooisland, southaustralia, sa, australia, australiansealion, 300mm	beach, sea, surf, strand, shore, wave, seascape, sand, ocean, waves
	<no text>	night, notte, traffic, light, lights, parking, darkness, lowlight, nacht, glow
	mickikrimmel, micipedia, headshot	portrait, girl, woman, lady, blonde, pretty, gorgeous, expression, model
	camera, jahdakine, lightpainting, relection, doublepaneglass, wowiekazowie	blue, art, artwork, artistic, surreal, expression, original, artist, gallery, patterns

# 자연어 지도학습(Natural language supervision) (2/2)

- 트랜스포머(Vaswani et al. ,2017) 개발 이후, 트랜스포머 기반 언어모델(BERT, GPT 등)이 방대한 텍스트 학습을 통해 복잡한 언어적 맥락을 고려한 자연어처리가 가능함을 보임.
- 이런 언어모델의 발전은 자연어 지도학습 기반 시각모델의 연구도 새로운 돌파구를 만들어냄.



# 대규모 *image-caption* 데이터셋의 구축(1/2)

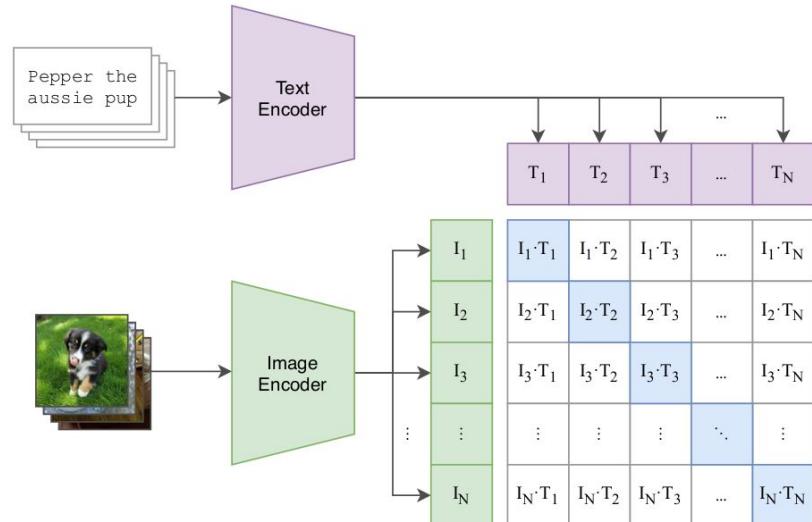
- 자연어 지도학습은 개념적으로 훌륭하지만 낮은 성능 때문에 실제로 사용하는 경우는 매우 드묾
- 최신 성능의 시각모델들은 지도학습 방식보다 데이터 확장성이 큰 **semi-supervised**, **self-supervised** 방법 등을 사용
  - 지도학습은 수백~수천만 규모의 데이터를 사용하는 반면 **semi-supervised**, **self-supervised** 방법은 수억~수십억 규모의 데이터를 사용해 사전학습 모델을 구축
- 자연어 지도학습을 위한 데이터셋 구축 필요
  - 기존의 자연어 지도학습이 테스트한 MS-COCO(Lin et al., 2014), Visual Genome (Krishna et al., 2017), YFCC100M (Thomee et al., 2016) 등은 십만 장 규모로 작고, YFCC100M은 1억장 수준이지만 자연어 레이블의 낮은 수준이 문제임.

## 대규모 *image-caption* 데이터셋의 구축(2/2)

- OpenAI 연구팀은 50만 개의 쿼리 리스트를 구성하고, 인터넷에서 각 쿼리와 관련한 4억 개의 데이터쌍(이미지, 캡션)을 수집함.
  - 다양한 단어로 수집한 이유는 최대한 다양한 시각적 개념을 학습하기 위함
  - 구축된 데이터셋은 GPT-2를 학습하는데 사용된 WebText 데이터셋과 비슷한 규모
- 쿼리 리스트는 영문 위키피디아에서 100회 이상 등장한 단어 50만 개로 구성
  - 기본 쿼리들은 다시 bi-gram을 사용해 이를 증강함. bi-gram은 자연어에서 두 단어 조합에 대한 확률모델로 증강 시 한 단어보다 더 특정한 의미를 포착할 수 있음.
  - 예를 들어, “고양이”와 “개”가 각각 기본 쿼리 리스트에 있다면 bi-gram에 의해 “고양이 개”가 새로운 쿼리가 추가될 수 있음.
  - 최대한 클래스 균형을 맞추기 위해 각 쿼리마다 2만 여개의 데이터쌍 수집

# CLIP : Contrastive Language-Image Pretraining

- CLIP(Contrastive Language-Image Pretraining)은 사진과 캡션 사이 연관성을 학습하는 자연어 지도학습
  - CLIP의 임베딩 공간에서 사진은 연관된 캡션과 가까워지고 무작위적인 캡션과 멀어짐
- OpenAI 연구팀은 4억 쌍(사진-캡션)의 웹데이터를 수집해 CLIP 모델의 사전학습에 사용



# CLIP : Contrastive Language-Image Pretraining

- Easy to scale
  - 전통적인 지도학습의 막대한 노동이 필요한 1-of-N 레이블링 방식보다 자연어 지도학습은 웹수집을 통한 데이터 규모의 확장이 상대적으로 쉬움.
- Language-image connection
  - 자율학습(self-supervised learning) 또는 비지도학습(unsupervised learning)과 달리 단순히 시각표현만 학습하는 것이 아닌, 시각표현과 언어를 연결한다는 강점 존재
- Zero-shot transferable
  - CLIP은 자연어를 시각적 패턴과 연결할 수 있기 때문에 언어로 문제를 정의해 추가적 학습 없이 문제를 푸는 zero shot transfer가 높은 정확도로 가능.

# 학습 방법론 개발(1/2)

- 연구초기 접근방법은 Virtex처럼 CNN과 text transformer를 결합해 초기값(scratch) 학습 채택
  - 간단한 **Bag-of-words** 모델을 쓰는 것보다도 효과적으로 확장하는데 어려움
  - 캡션의 단어들을 정확히 예측하려고 하는 방식이 너무 어렵기 때문
- Contrastive learning 사용
  - 이미지의 아웃풋과 캡션이 동일하도록 예측하는 대신 **contrastive learning**이 더 효과적

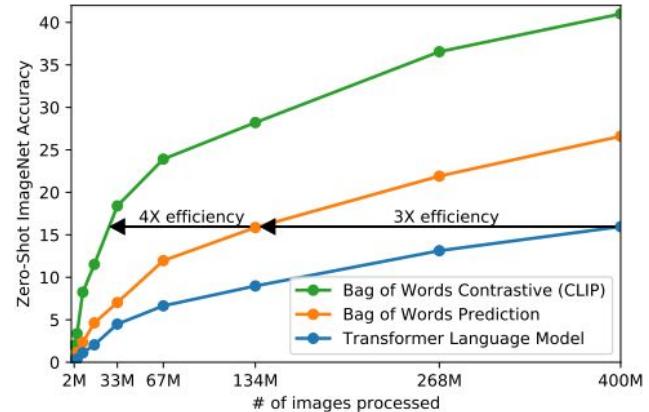
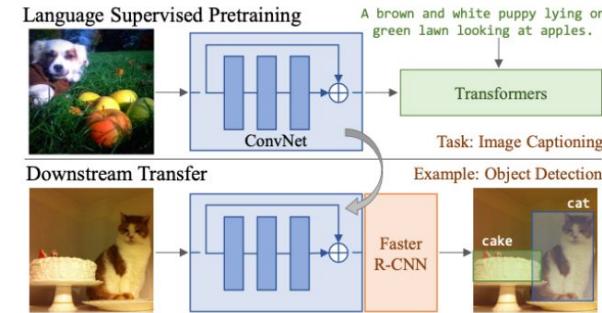


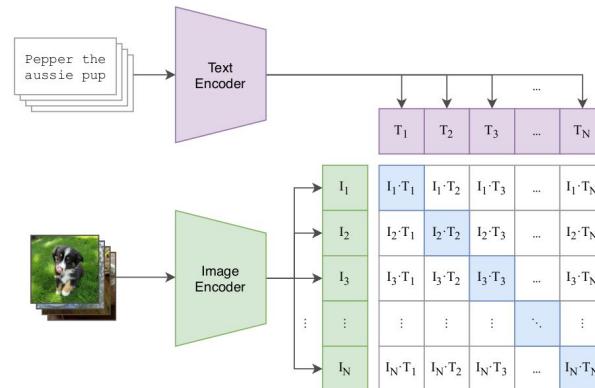
그림. Transformer Language Model(Virtex)의 낮은 확장성

# 학습 방법론 개발(2/2)

- 연구초기 접근방법은 Virtex처럼 CNN과 text transformer를 결합해 초기값(scratch) 학습 채택
  - 간단한 **Bag-of-words** 모델을 쓰는 것보다도 성능향상이 어려움
  - 이유는 캡션의 단어들을 정확히 예측하려고 하는 방식이 학습을 어렵게 만듦.
- Contrastive learning이 새로운 대안으로 여겨짐
  - Tian et al., 2019 에 따르면 캡션을 직접 예측하는 방식보다 캡션이 영상과 연관있는가를 예측하는 것이 더 효과적

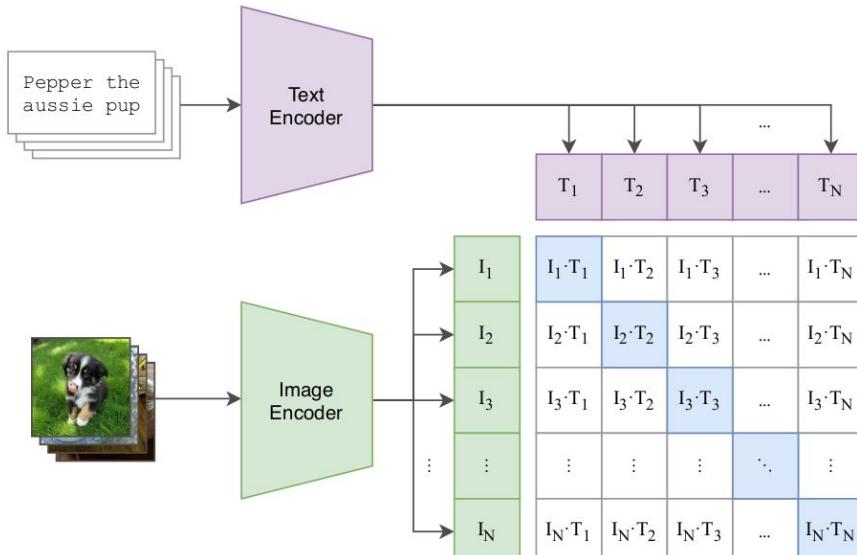


그림A. Equivalent predictive objective method(Vortex)



그림B. Contrastive learning method(CLIP)

# 학습 알고리즘(1/3)



그림A. N batch contrastive learning

```

# image_encoder - ResNet or Vision Transformer
# text_encoder - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l] - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T) #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

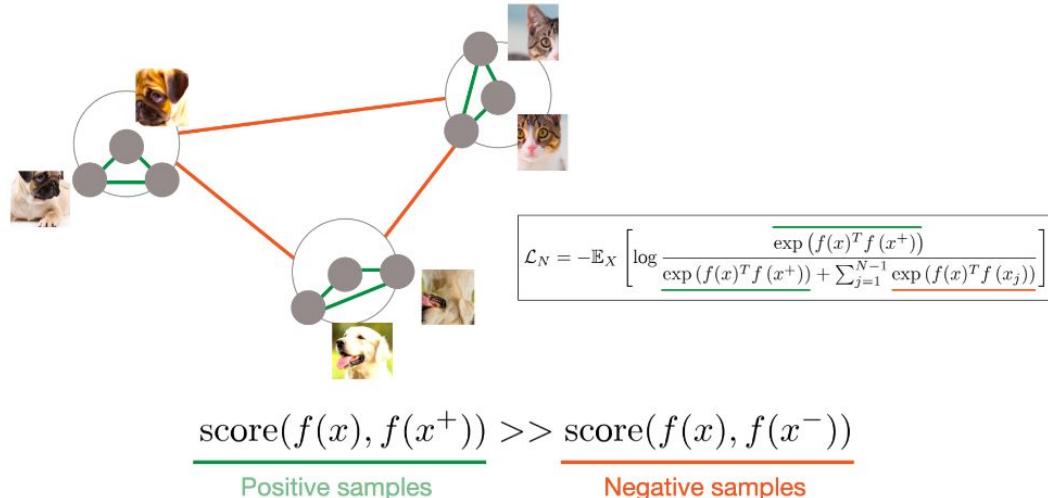
# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss = (loss_i + loss_t)/2

```

그림B. training pseudo code

# 학습 알고리즘(2/3)

- CLIP은 N 개의 배치사이즈에서  $N \times N$ 의 비교쌍(영상, 캡션)을 학습
  - $N \times N$  비교쌍 중 긍정쌍은 N개 부정쌍은  $N^{**2} - N$ 개 생성
  - 영상 인코더와 텍스트 인코더는 비교쌍 중 긍정쌍의 코사인 유사도를 최대화, 부정쌍은 최소화함
  - 학습의 결과로 영상 인코더와 텍스트 인코더는 하나의 **multi-modal** 임베딩 공간에서 연결됨



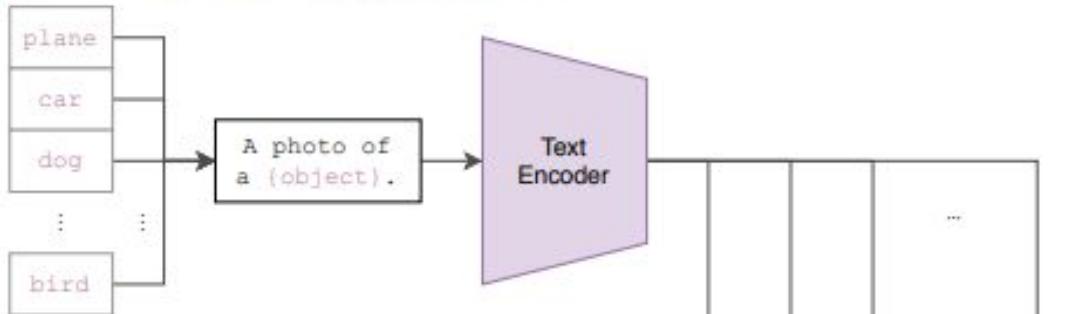
## 학습 알고리즘(3/3)

---

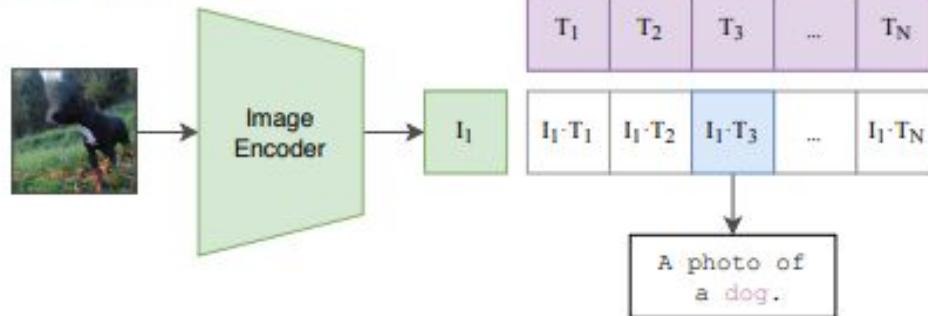
- 영상 인코더는 scratch, 텍스트 인코더는 pre-trained weights 사용
- 영상 인코더는 ResNet과 ViT를 사용
  - ResNet-50의 global average pooling은 attention pooling으로 변경
  - 3가지 타입 : ResNet-50, ResNet-101, EfficientNet-style scaling ResNet 테스트
- 텍스트 인코더는 Transformer 사용
  - 63M-parameter 12-layer 512-wide model with 8 attention heads.
  - 49,152 vocab size
  - 3가지 타입 : ViT-B/32, ViT-B/16, ViT-L/14
- during 32 epochs, train with Adam optimizer, decoupled weight decay, cosine learning rate scheduler, mini-batch 32,768

# Zero-shot transferable

Create dataset classifier from label text



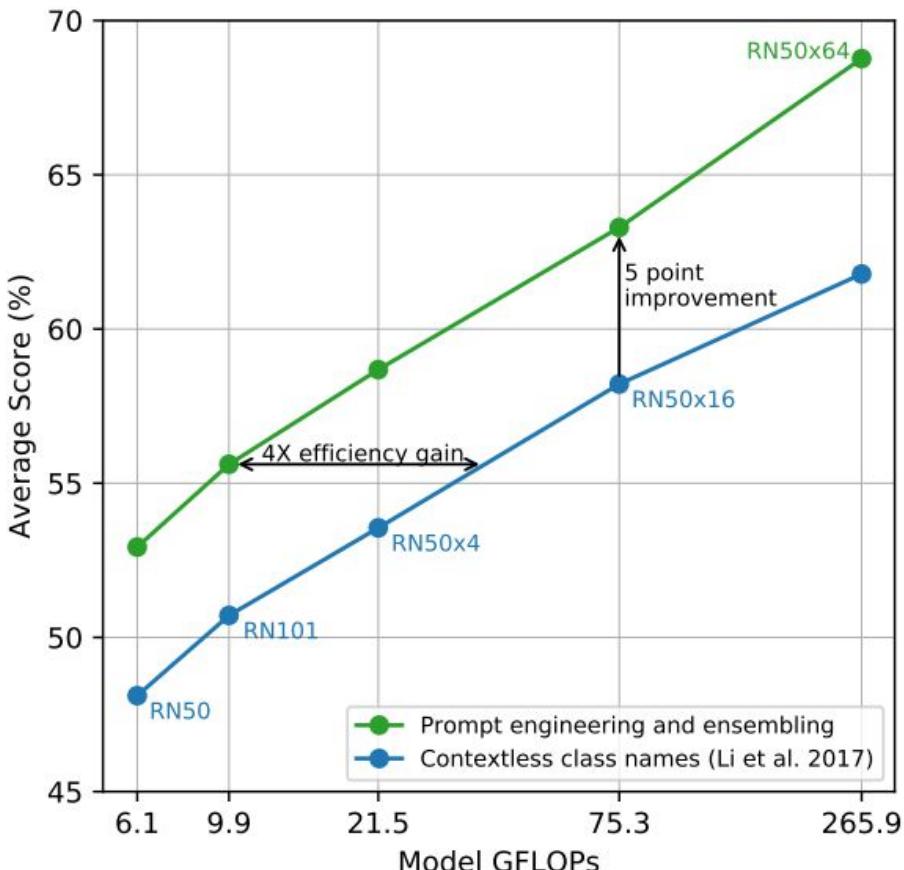
Use for zero-shot prediction



	aYahoo	ImageNet	SUN
Visual N-Grams	72.4	11.5	23.0
CLIP	<b>98.4</b>	<b>76.2</b>	<b>58.5</b>

Table 1. Comparing CLIP to prior zero-shot transfer image classification results. CLIP improves performance on all three datasets by a large amount. This improvement reflects many differences in the 4 years since the development of Visual N-Grams (Li et al., 2017).

# Zero-shot transferable

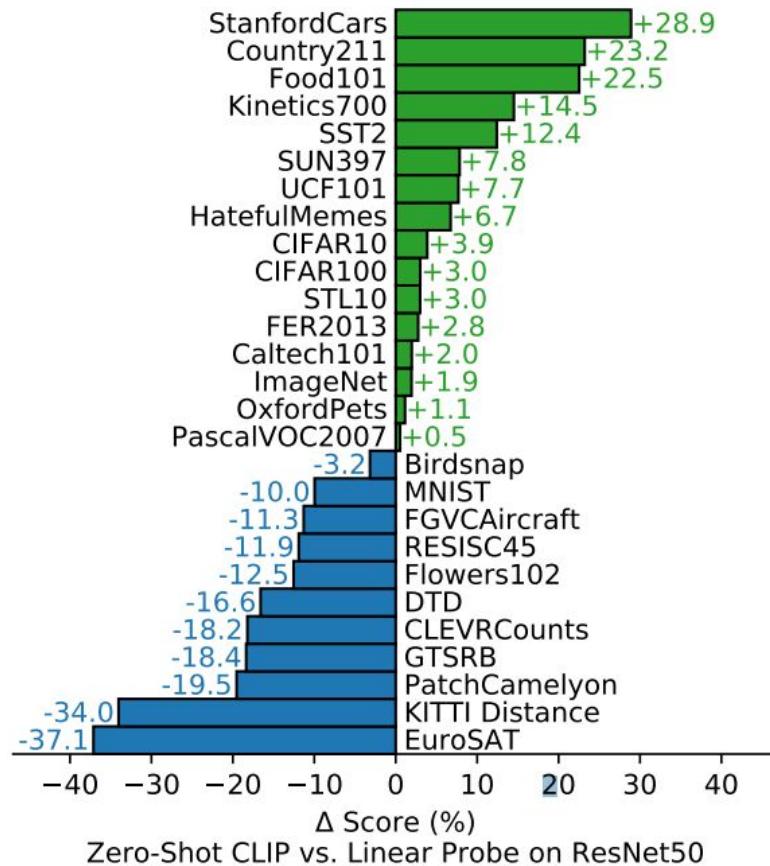


**Figure 4. Prompt engineering and ensembling improve zero-shot performance.** Compared to the baseline of using contextless class names, prompt engineering and ensembling boost zero-shot classification performance by almost 5 points on average across 36 datasets. This improvement is similar to the gain from using 4 times more compute with the baseline zero-shot method but is “free” when amortized over many predictions.

다의성 문제(건설 기계 Crane과 날아다니는 Crane), 캡션의 형식에 따라 달라지는 성능...

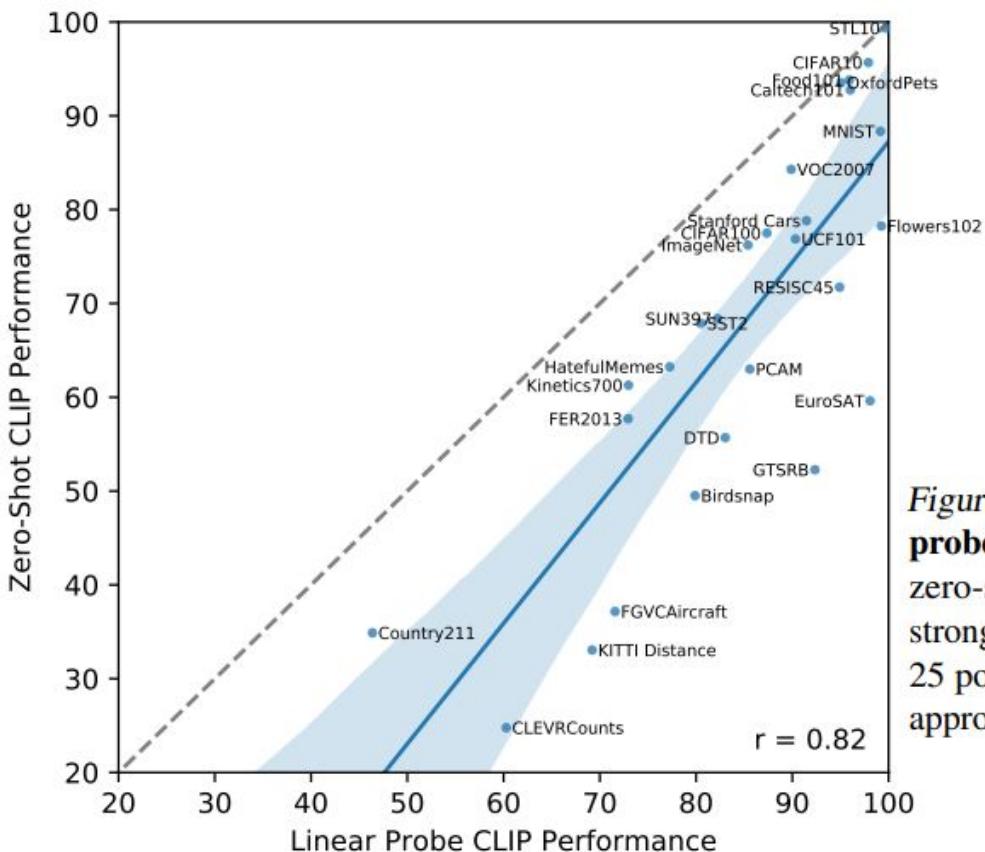
프롬프트 엔지니어링이 상당히 중요

# Zero-shot transferable



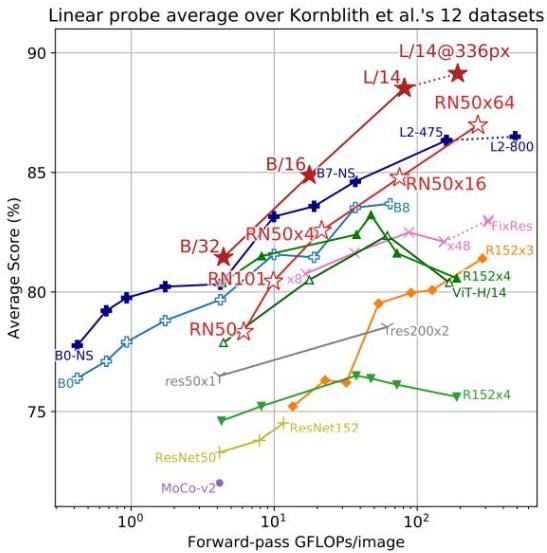
**Figure 5. Zero-shot CLIP is competitive with a fully supervised baseline.** Across a 27 dataset eval suite, a zero-shot CLIP classifier outperforms a fully supervised linear classifier fitted on ResNet-50 features on 16 datasets, including ImageNet.

# Zero-shot transferable



**Figure 8. Zero-shot performance is correlated with linear probe performance but still mostly sub-optimal.** Comparing zero-shot and linear probe performance across datasets shows a strong correlation with zero-shot performance mostly shifted 10 to 25 points lower. On only 5 datasets does zero-shot performance approach linear probe performance ( $\leq 3$  point difference).

# Performance



★ CLIP-ViT	✗ Instagram-pretrained	△ ViT (ImageNet-21k)
★ CLIP-ResNet	◆ SimCLRv2	▲ BiT-M
■ EfficientNet-NoisyStudent	▲ BYOL	▲ BiT-S
■ EfficientNet	○ MoCo	✚ ResNet

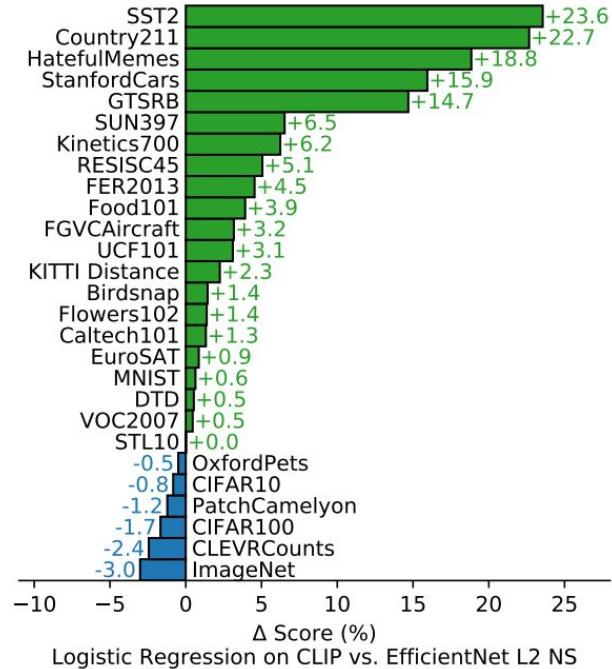
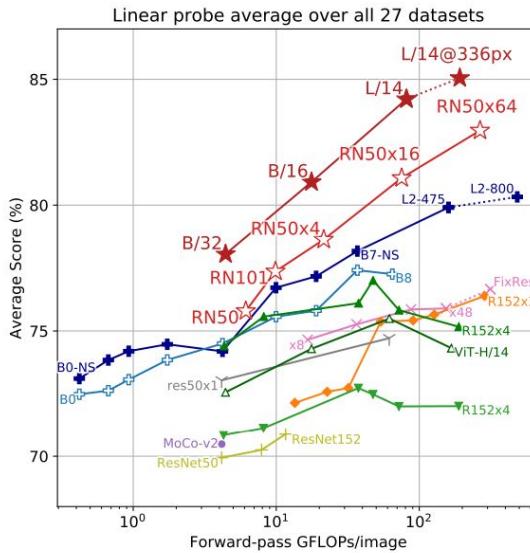
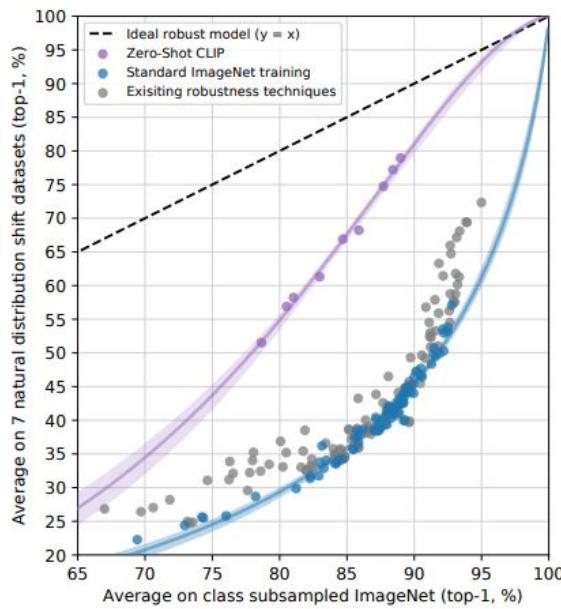


Figure 10. Linear probe performance of CLIP models in comparison with state-of-the-art computer vision models, including EfficientNet (Tan & Le, 2019; Xie et al., 2020), MoCo (Chen et al., 2020d), Instagram-pretrained ResNeXt models (Mahajan et al., 2018; Touvron et al., 2019), BiT (Kolesnikov et al., 2019), ViT (Dosovitskiy et al., 2020), SimCLRv2 (Chen et al., 2020c), BYOL (Grill et al., 2020), and the original ResNet models (He et al., 2016b). (Left) Scores are averaged over 12 datasets studied by Kornblith et al. (2019). (Right) Scores are averaged over 27 datasets that contain a wider variety of distributions. Dotted lines indicate models fine-tuned or evaluated on images at a higher-resolution than pre-training. See Table 10 for individual scores and Figure 20 for plots for each dataset.

Figure 11. CLIP's features outperform the features of the best ImageNet model on a wide variety of datasets. Fitting a linear classifier on CLIP's features outperforms using the Noisy Student EfficientNet-L2 on 21 out of 27 datasets.

# Performance



**Figure 13. Zero-shot CLIP is much more robust to distribution shift than standard ImageNet models.** (Left) An ideal robust model (dashed line) performs equally well on the ImageNet distribution and on other natural image distributions. Zero-shot CLIP models shrink this “robustness gap” by up to 75%. Linear fits on logit transformed values are shown with bootstrap estimated 95% confidence intervals. (Right) Visualizing distribution shift for bananas, a class shared across 5 of the 7 natural distribution shift datasets. The performance of the best zero-shot CLIP model, ViT-L/14@336px, is compared with a model that has the same performance on the ImageNet validation set, ResNet-101.

# Learning to Prompt for Vision-Language Models

Kaiyang Zhou Jingkang Yang Chen Change Loy Ziwei Liu  
S-Lab, Nanyang Technological University, Singapore

- *new question: How should we represent our classes for a given problem in natural language?*
- Zhou et al. 2020의 Prompt learning to vision domain
  - CLIP 같은 방대한 데이터를 학습한 거대모델을 문제에 맞게 재학습하는 것은 비현실적이며, 잘 학습된 표상공간을 손상시킬 수 있음
  - 대신 “a photo of a {class}, a type of pet”의 “a type of pet”과 같이 성능 개선에 도움이 되는 컨텍스트를 찾는 프롬프트 엔지니어링이 사용됐으나 매우 비효율적이고 시행착오적이라는 단점 존재
  - Zhou et al. 은 CoOp(Context Optimization)을 제안해 프롬프트 안의 새로운 컨텍스트 단어를 학습 가능한 벡터의 집합으로 변환하여 적은 수의 학습용 사진으로 새로운 개념을 학습하는 프롬프트 학습 제안

# Learning to Prompt for Vision-Language Models

Kaiyang Zhou Jingkang Yang Chen Change Loy Ziwei Liu  
S-Lab, Nanyang Technological University, Singapore

Caltech101	Prompt	Accuracy
	a [CLASS].	82.68
	a photo of [CLASS].	80.81
	a photo of a [CLASS].	86.29
	[V] <sub>1</sub> [V] <sub>2</sub> ... [V] <sub>M</sub> [CLASS].	<b>91.83</b>

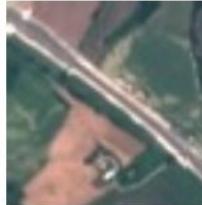
(a)

Flowers102	Prompt	Accuracy
	a photo of a [CLASS].	60.86
	a flower photo of a [CLASS].	65.81
	a photo of a [CLASS], a type of flower.	66.14
	[V] <sub>1</sub> [V] <sub>2</sub> ... [V] <sub>M</sub> [CLASS].	<b>94.51</b>

(b)

Describable Textures (DTD)	Prompt	Accuracy
	a photo of a [CLASS].	39.83
	a photo of a [CLASS] texture.	40.25
	[CLASS] texture.	42.32
	[V] <sub>1</sub> [V] <sub>2</sub> ... [V] <sub>M</sub> [CLASS].	<b>63.58</b>

(c)

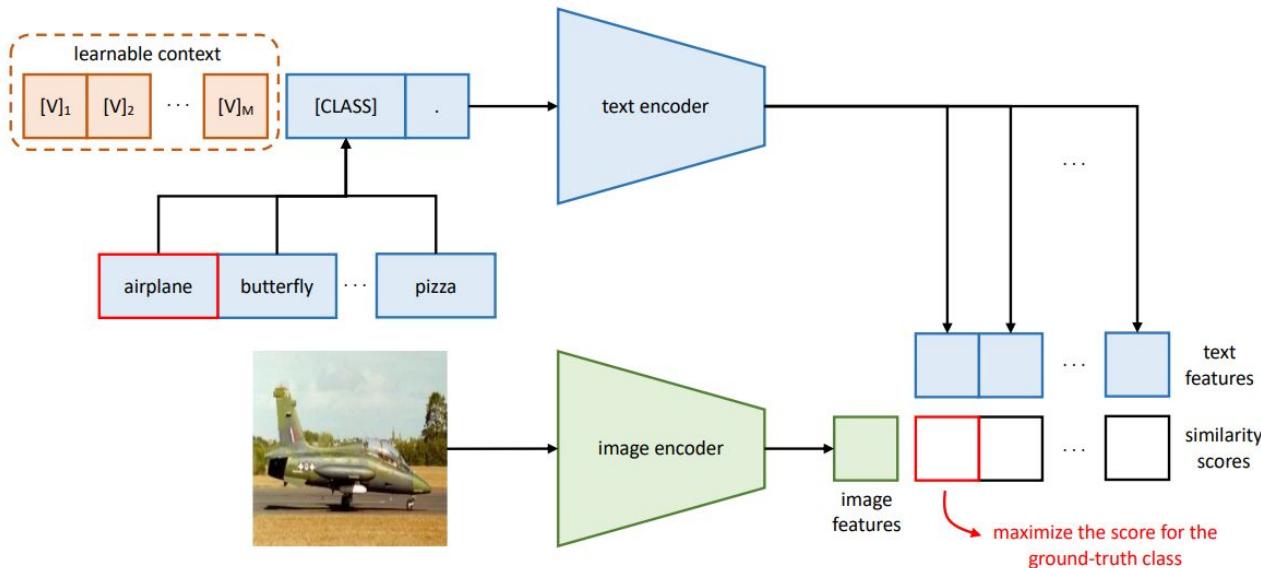
EuroSAT	Prompt	Accuracy
	a photo of a [CLASS].	24.17
	a satellite photo of [CLASS].	37.46
	a centered satellite photo of [CLASS].	37.56
	[V] <sub>1</sub> [V] <sub>2</sub> ... [V] <sub>M</sub> [CLASS].	<b>83.53</b>

(d)

**Fig. 1 Prompt engineering vs Context Optimization (CoOp).** The former needs to use a held-out validation set for words tuning, which is inefficient; the latter automates the process and requires only a few labeled images for learning.

# Learning to Prompt for Vision-Language Models

Kaiyang Zhou Jingkang Yang Chen Change Loy Ziwei Liu  
S-Lab, Nanyang Technological University, Singapore



**Fig. 2 Overview of Context Optimization (CoOp).** The main idea is to model a prompt's context using a set of learnable vectors, which can be optimized through minimizing the classification loss. Two designs are proposed: one is unified context, which shares the same context vectors with all classes; and the other is class-specific context, which learns for each class a specific set of context vectors.

# Learning to Prompt for Vision-Language Models

Kaiyang Zhou Jingkang Yang Chen Change Loy Ziwei Liu  
S-Lab, Nanyang Technological University, Singapore

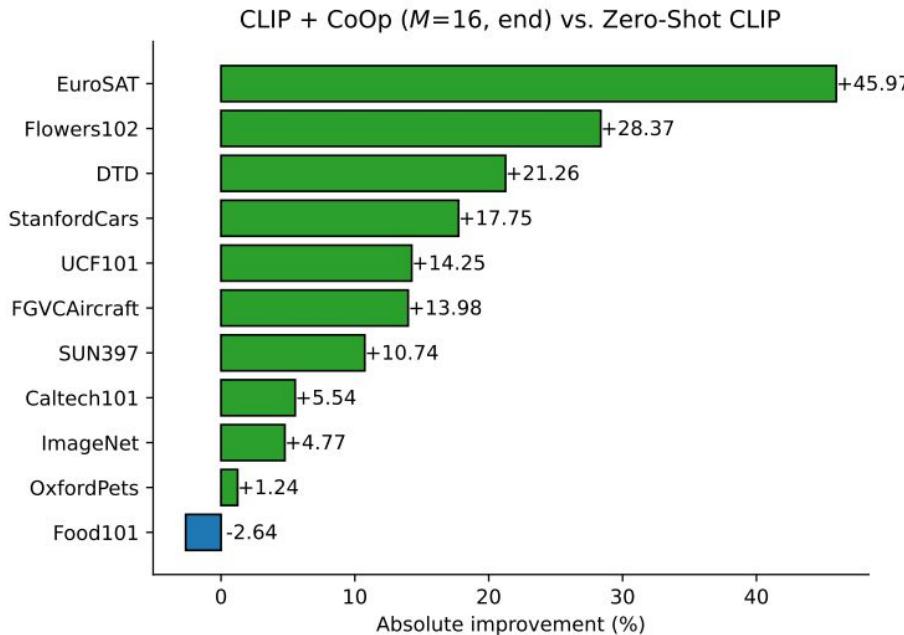


Fig. 4 Comparison with hand-crafted prompts.

Method	Source	Target			
		-V2	-Sketch	-A	-R
<b>ResNet-50</b>					
Zero-Shot CLIP	58.18	51.34	33.32	21.65	56.00
Linear Probe CLIP	55.87	45.97	19.07	12.74	34.86
CLIP + CoOp ( $M=16$ )	62.95	55.11	32.74	22.12	54.96
CLIP + CoOp ( $M=4$ )	<b>63.33</b>	<b>55.40</b>	<b>34.67</b>	<b>23.06</b>	<b>56.60</b>
<b>ResNet-101</b>					
Zero-Shot CLIP	61.62	54.81	38.71	28.05	64.38
Linear Probe CLIP	59.75	50.05	26.80	19.44	47.19
CLIP + CoOp ( $M=16$ )	<b>66.60</b>	<b>58.66</b>	39.08	28.89	63.00
CLIP + CoOp ( $M=4$ )	65.98	58.60	<b>40.40</b>	<b>29.60</b>	<b>64.98</b>
<b>ViT-B/32</b>					
Zero-Shot CLIP	62.05	54.79	40.82	29.57	<b>65.99</b>
Linear Probe CLIP	59.58	49.73	28.06	19.67	47.20
CLIP + CoOp ( $M=16$ )	<b>66.85</b>	58.08	40.44	30.62	64.45
CLIP + CoOp ( $M=4$ )	66.34	<b>58.24</b>	<b>41.48</b>	<b>31.34</b>	65.78
<b>ViT-B/16</b>					
Zero-Shot CLIP	66.73	60.83	46.15	47.77	73.96
Linear Probe CLIP	65.85	56.26	34.77	35.68	58.43
CLIP + CoOp ( $M=16$ )	<b>71.92</b>	64.18	46.71	48.41	74.32
CLIP + CoOp ( $M=4$ )	71.73	<b>64.56</b>	<b>47.89</b>	<b>49.93</b>	<b>75.14</b>

M : CoOp's context length

# Conditional prompt learning for Vision-Language Models

Kaiyang Zhou Jingkang Yang Chen Change Loy Ziwei Liu  
S-Lab, Nanyang Technological University, Singapore

- 저자들은 CoOp이 새로 배운 컨텍스트가 학습데이터에서 보이지 않은 다른 클래스들에 일반화되지 않는 치명적인 문제를 발견하고 이를 해결하는 CoCoOp(Conditional Context Optimization)을 제안함.

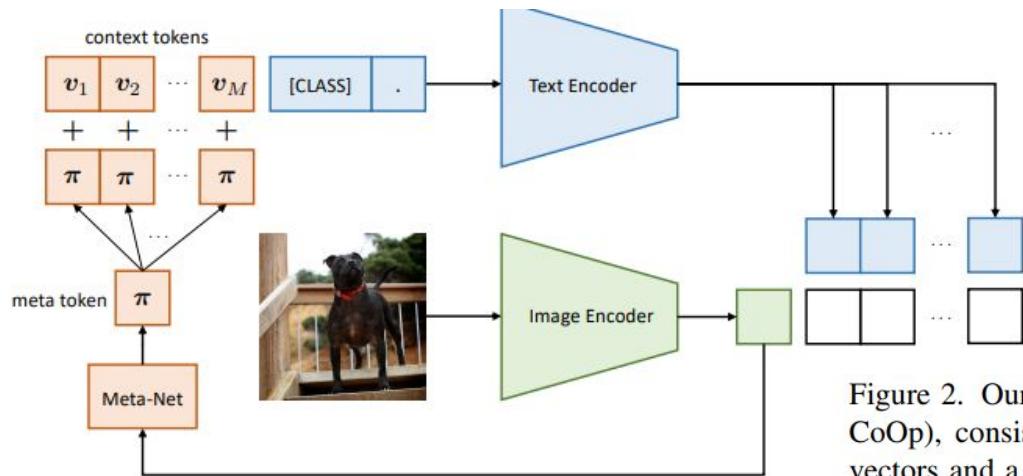
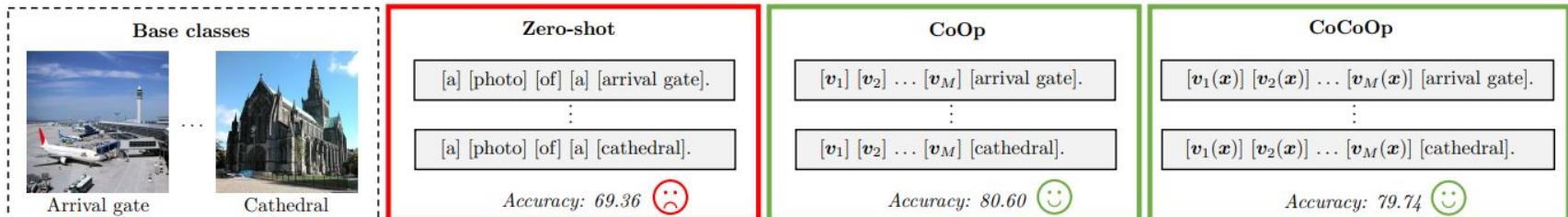


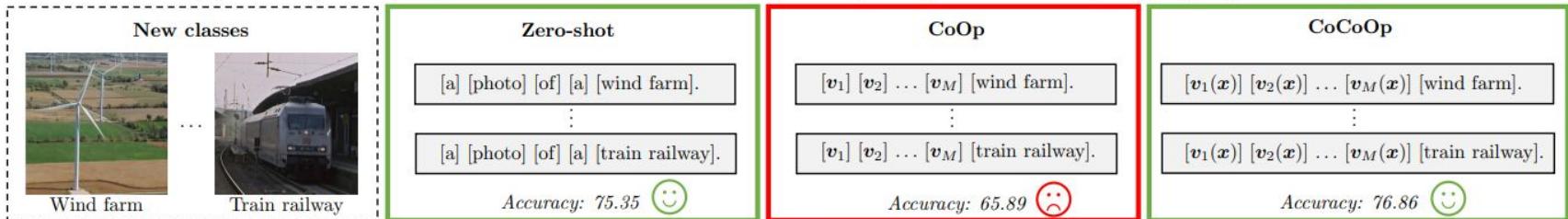
Figure 2. Our approach, Conditional Context Optimization (CoCoOp), consists of two learnable components: a set of context vectors and a lightweight neural network (Meta-Net) that generates for each image an input-conditional token.

# Conditional prompt learning for Vision-Language Models

Kaiyang Zhou Jingkang Yang Chen Change Loy Ziwei Liu  
S-Lab, Nanyang Technological University, Singapore



(a) Both CoOp and CoCoOp work well on the base classes observed during training and beat manual prompts by a significant margin.



(b) The instance-conditional prompts learned by CoCoOp are much more generalizable than CoOp to the unseen classes.

Figure 1. **Motivation of our research: to learn generalizable prompts.** The images are randomly selected from SUN397 [55], which is a widely-used scene recognition dataset.

# Deep generative learning

- Introduction to generative learning
- VAE, GAN, Flow-based

# Introduction to deep generative learning?



Train

A thick green arrow pointing from left to right, indicating the direction of training.

Samples from a Data Distribution

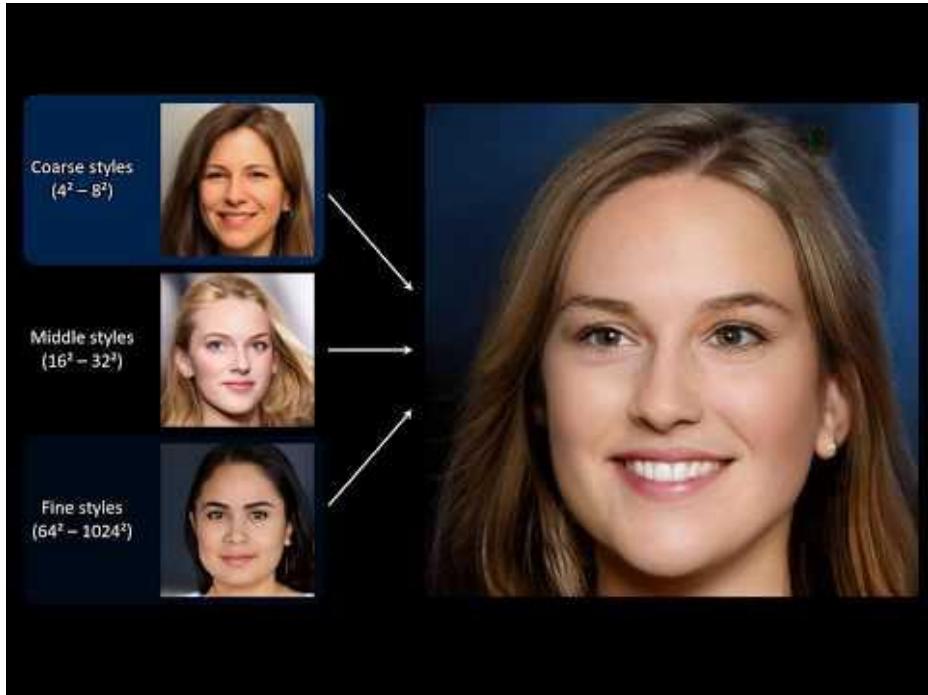
Neural Network



Sample

A thick green arrow pointing from left to right, indicating the direction of sampling.

# *Applications(1)*



NVIDIA StyleGAN

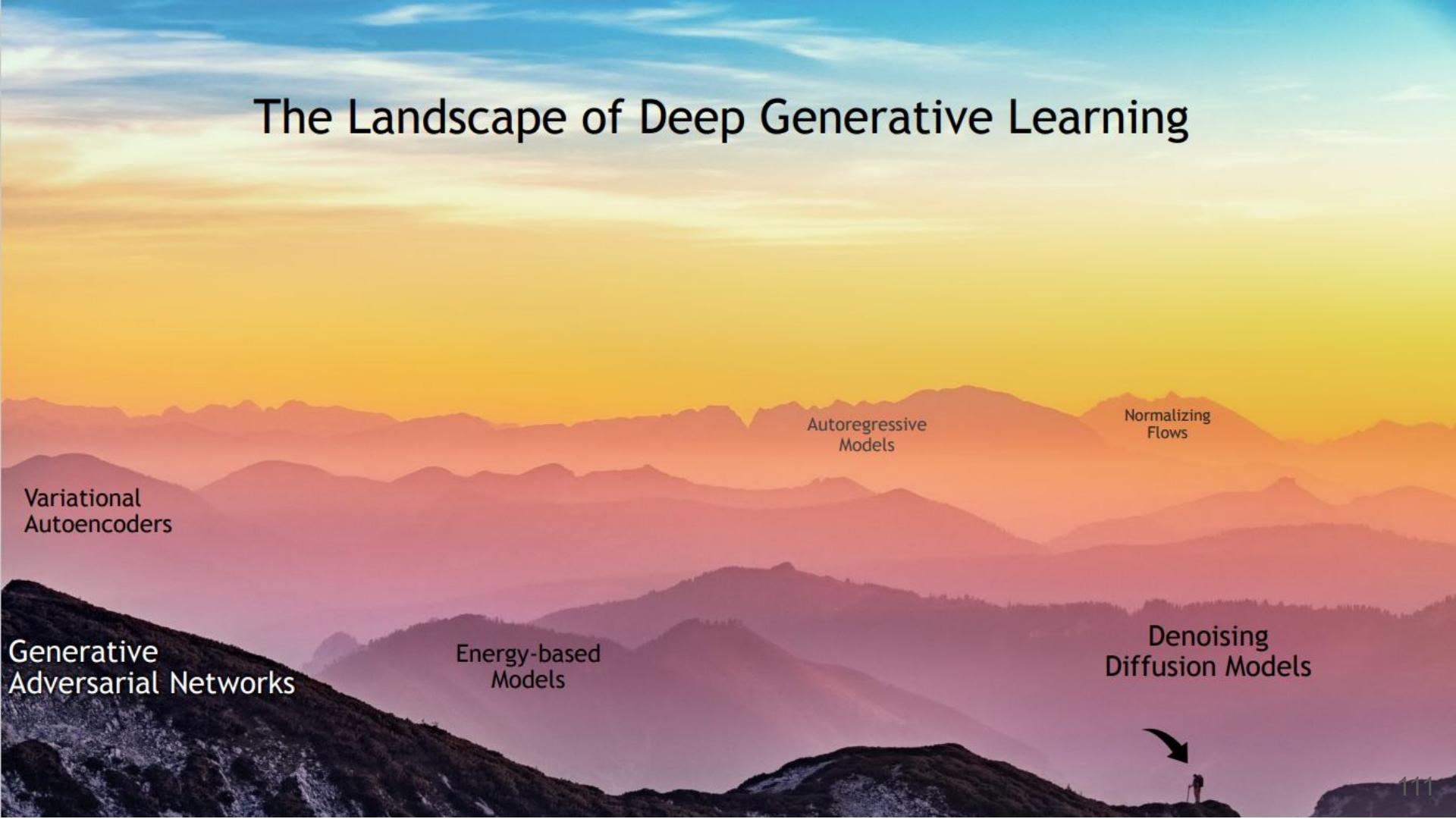


GoogleAI Imagen

## *Applications(2)*



# The Landscape of Deep Generative Learning

The background of the slide features a wide-angle photograph of a mountain range at sunset. The sky is a gradient from blue at the top to orange and yellow near the horizon. In the foreground, the dark silhouette of a mountain slope is visible on the left. On the right side, a small figure of a person stands on a rocky outcrop, holding a long pole or flag that points towards the center of the slide. The overall atmosphere is serene and vast.

Variational  
Autoencoders

Generative  
Adversarial Networks

Energy-based  
Models

Autoregressive  
Models

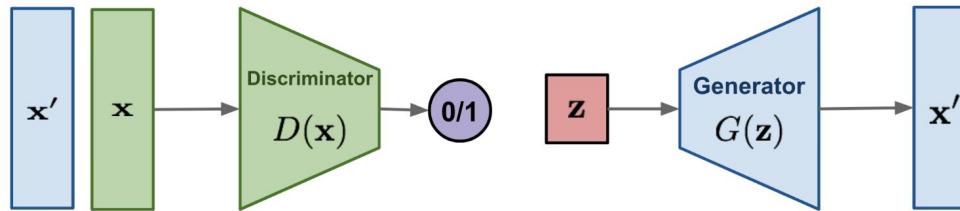
Normalizing  
Flows

Denoising  
Diffusion Models

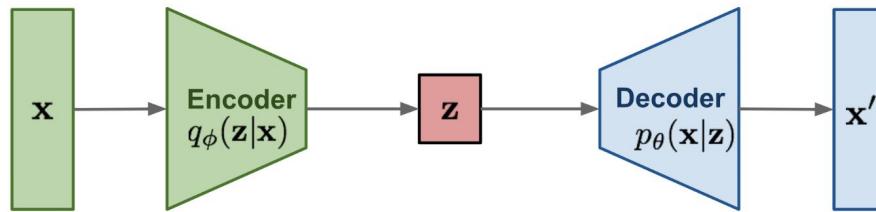


# Three types of Generative models

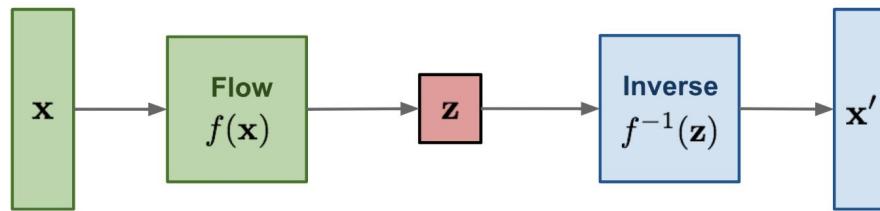
**GAN:** minimize the classification error loss.



**VAE:** maximize ELBO.

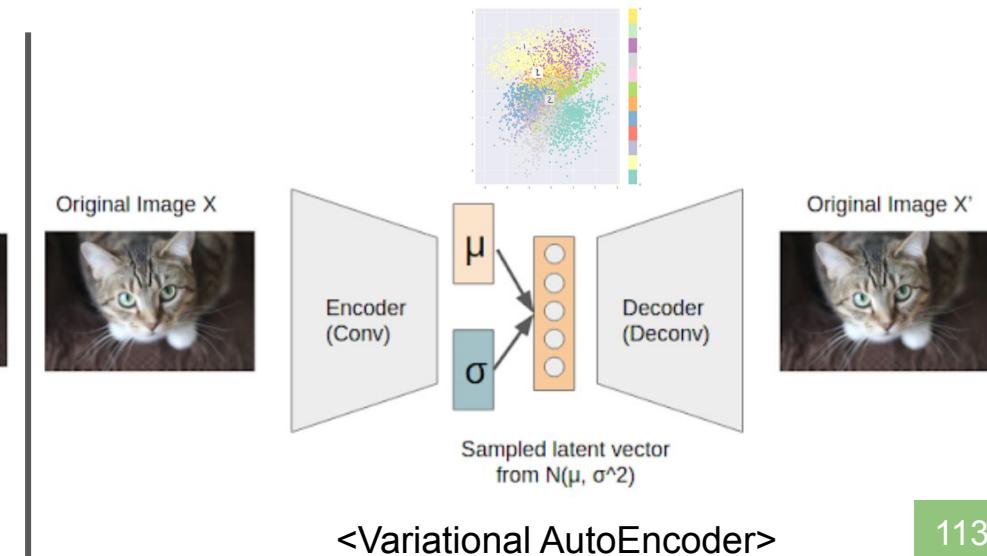
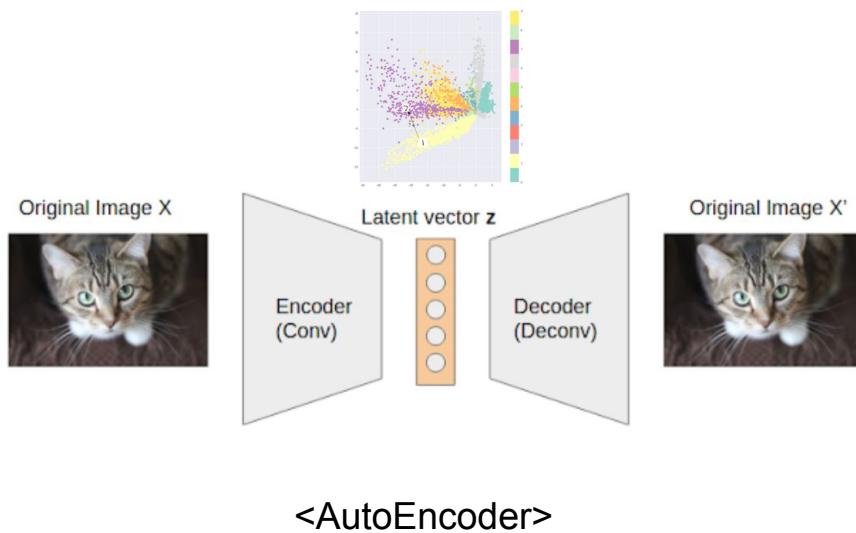


**Flow-based generative models:** minimize the negative log-likelihood



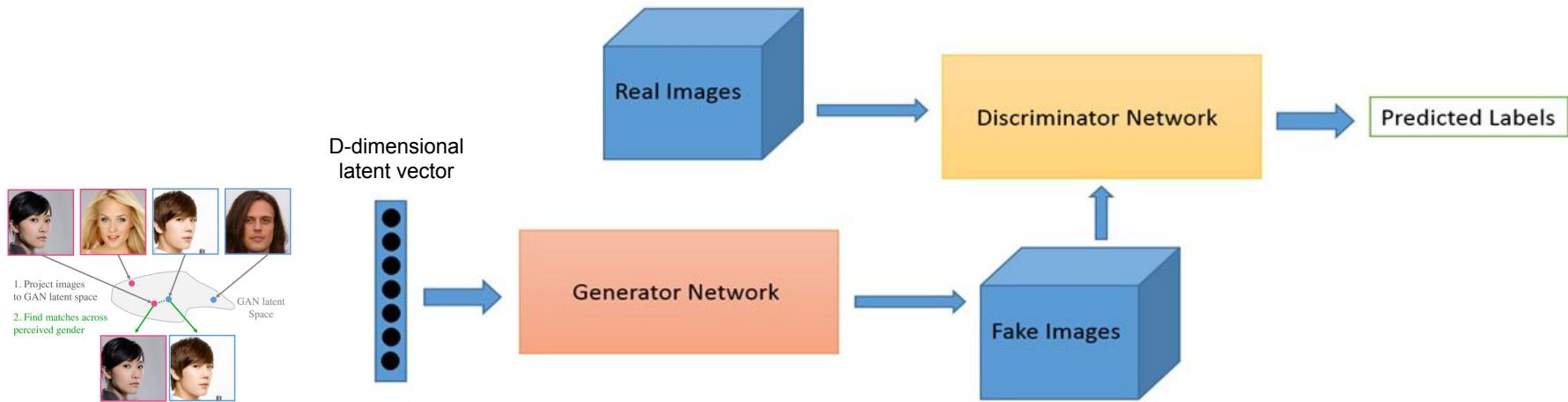
# 1. VAE(Variational AutoEncoders)

- AutoEncoder는 비지도 기반 복원 학습을 통한 특징 학습 모델로, 데이터에 대한 차원축소를 강제해 요약된 latent vector를 생성하도록 학습.
- 반면, VAE의 encoder는 latent vector를 추출하는 대신 다변수 가우시안 분포를 예측해, 분포로부터 latent vector를 샘플링하도록 변경.
- 이미지의 좋은 representation을 학습하지만, 낮은 생성 품질이 단점

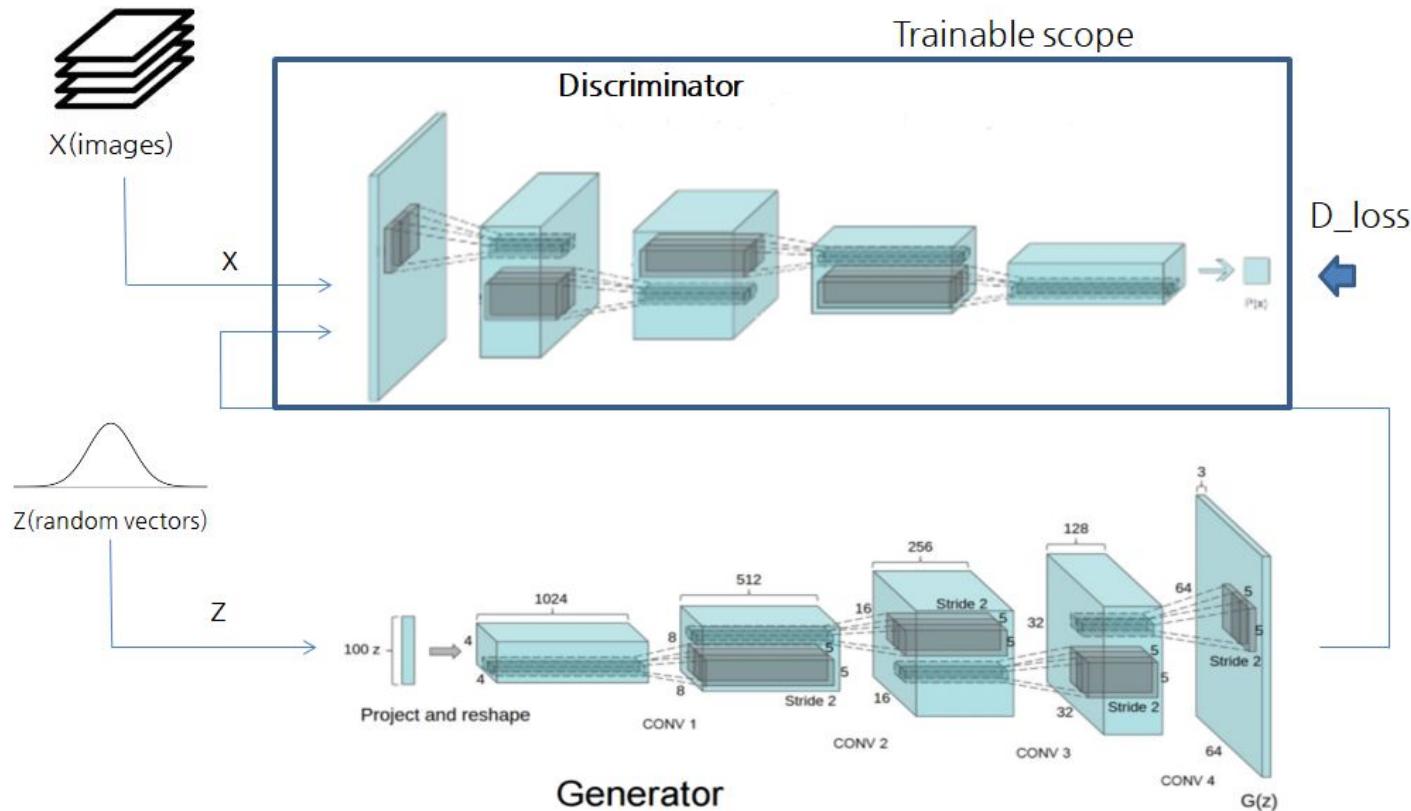


## 2. GAN(Generative Adversarial Networks)

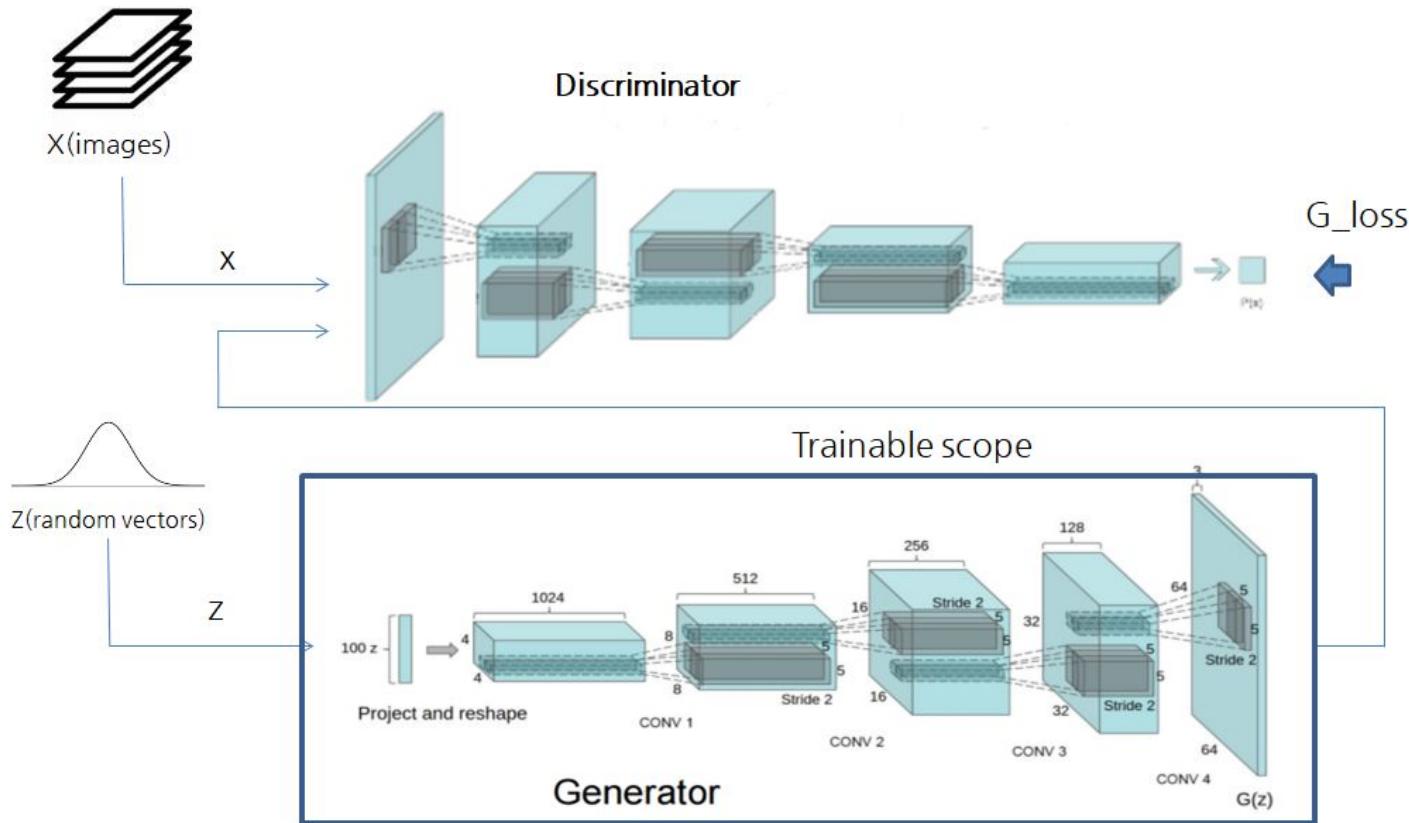
- GAN은 절대적 비지도 학습 기반의 생성모델이다.
- 분별자(Discriminator)는 실제 데이터와 가짜 데이터를 구분해 생성된 데이터를 분별하도록 학습.
- 생성자(Generator)는 noise vector로 가짜 데이터를 생성해 분별자를 속이도록 학습.



# Adversarial learning on Discriminator

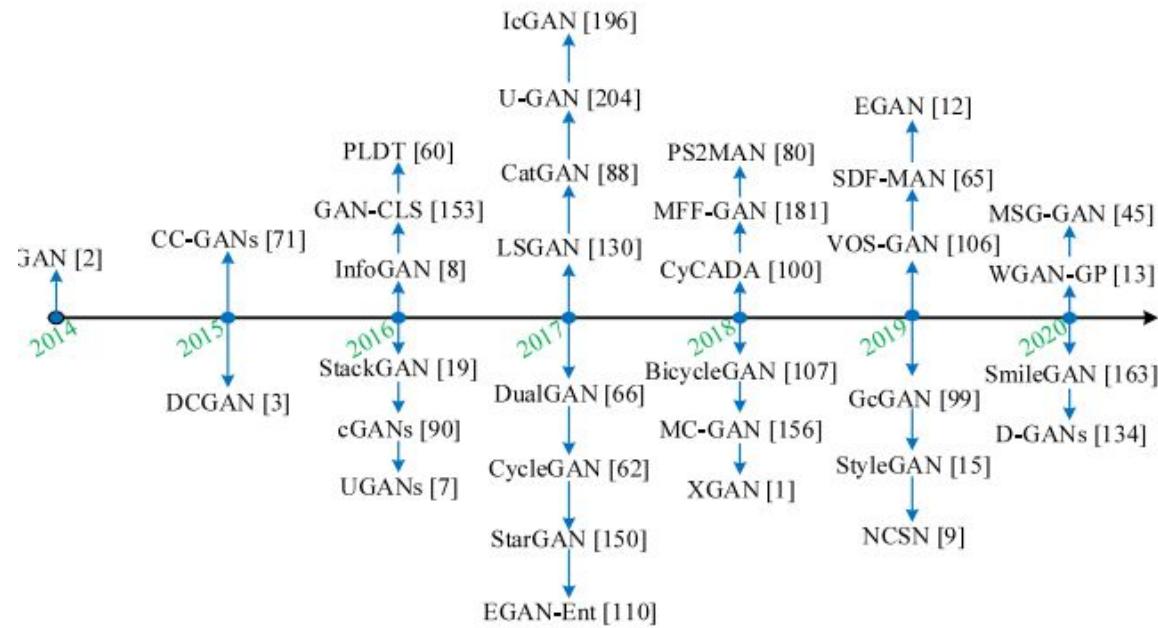


# Adversarial learning on Generator



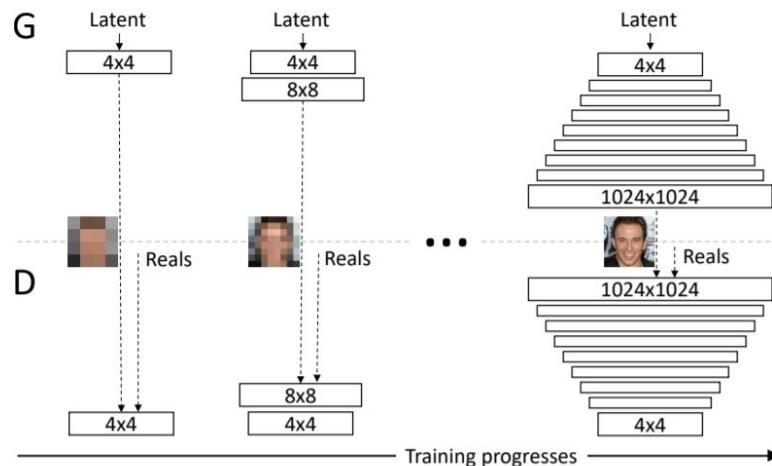
# History for GAN

- GAN은 가장 성공적인 생성 모델로 집중적인 연구들을 통한 많은 진보들이 있었음.
- 학습안정성, 고해상도 생성, 제어가능한 생성, 데이터 증강 등이 주요 연구주제



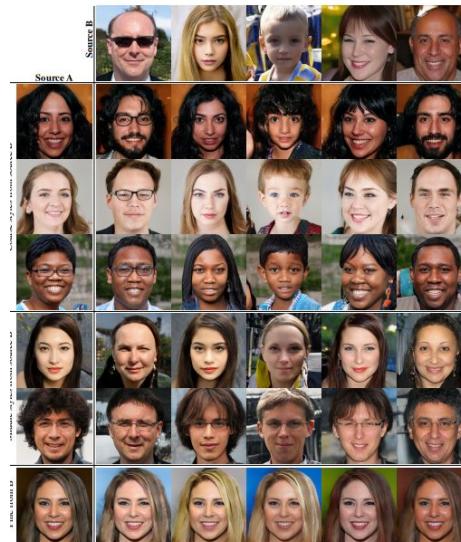
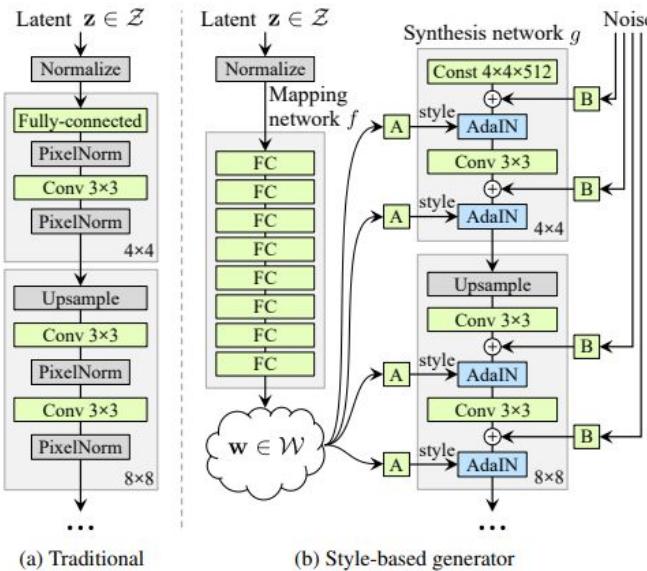
# PGGAN(Progressive Growing GAN)

- 고해상도 사진은 매우 고차원 pixel space에 분포함. 문제는 Latent space와 고차원 pixel space의 매핑 함수인 생성자의 학습은 기하급수적으로 어려워짐.
- 따라서 고화질이 될수록 생성자는 분별자를 속이기 힘들어지기에, GAN의 학습은 매우 느려지며 불완전해짐.
- 이를 해결하기 위해 PGGAN이 한 것은, 먼저 낮은 해상도로 학습을 진행하고, 점진적으로 더 높은 해상도로 학습을 하길 반복하는 형식을 사용.



# StyleGAN

- StyleGAN은 PGGAN을 기반으로  $(4 \times 4) \Rightarrow (8 \times 8) \dots \Rightarrow (1024 \times 1024)$  까지 점진적으로 해상도 증가
- 생성자의 전통적인 latent vector를 제거됨. 대신 Mapping network의 출력이 Synthesis network 중간중간의 AdaIN에서 스타일 특징을 결정함.



## Coarse style

*resolution of (4x4 – 8x8)*  
affects pose, general hair style, face shape, etc

## Middle style

*resolution of (16x16 – 32x32)*  
affects finer facial features, hair style, eyes open/closed, etc.

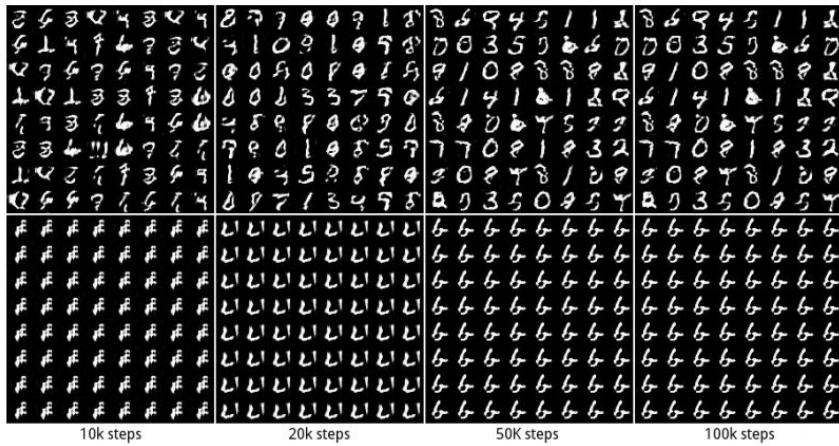
## Fine style –

*resolution of (64x64 – 1024x1024)* affects colors for (eye, hair and skin) and micro features.

← Style mixing  
in StyleGAN

# Unstable learning problems in GAN

- **Mode collapse** : 만약 생성자가 초기에 특정 데이터 유형을 온전하게 생성할 수 있다면, 오직 그 데이터 유형으로 분별자를 속일 수 있으며, 실제 데이터 분포가 갖는 다양성을 반영하지 못 함.
- **Unstability** : GANs은 학습의 수렴이 안정적이지 못한 문제가 존재. 특히 분별자의 성능이 생성자의 성능을 압도할 때, 생성자의 학습이 vanishing gradient에 의해 실패할 수 있음.

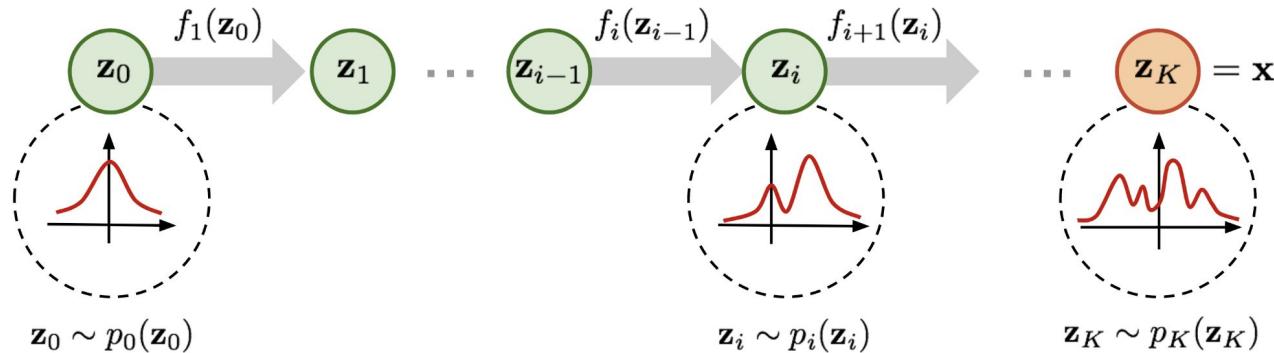


⇐ 정상적인 mnist 생성

⇐ mode collapse에 의한 비정상적 mnist 생성

### 3. Flow-based generative models

- Flow-based generative models은 연속적인 변환들로 구성
- 다른 두 모델과 달리 직접적으로 데이터 분포  $p(x)$ 를 학습.
- 직접적인  $p(x)$ 의 추정은 어렵기 때문에, 아래와 같이 연속적인 확률밀도 추정을 통해 다음 확률변수를 샘플링하는 방식으로 latent variable에서 데이터를 생성하도록 학습
- Normalizing flows, Autoregressive models, Diffusion models 등이 해당.
- 학습의 안정성과 좋은 퀄리티의 데이터를 생성할 수 있지만 느린 데이터 샘플링이 단점



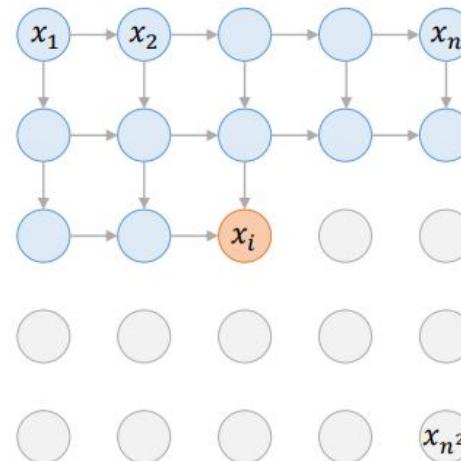
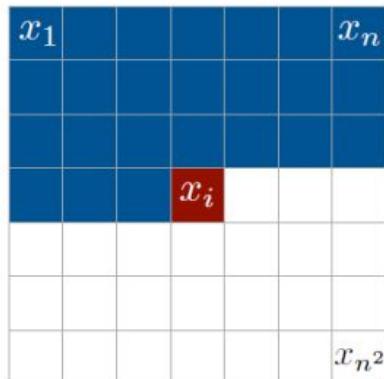
$$\mathbf{x} = \mathbf{z}_K = f_K \circ f_{K-1} \circ \cdots \circ f_1(\mathbf{z}_0)$$

# PixelRNN and PixelCNN(1)

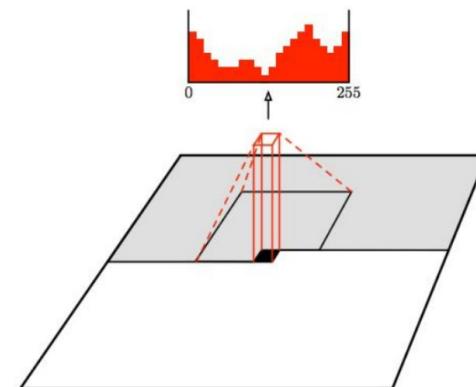
- 자기자신을 입력으로 하여 자기 자신을 예측하는 Autoregressive model(자기회귀모델)
- 이전의 픽셀값들로부터 다음 픽셀의 확률분포를 추정.
- 자기회귀에 의한 연속적인 픽셀값 추정으로 영상 생성

$$p(x) = \prod_{i=1}^n p(x_i|x_1, \dots, x_{i-1})$$

↑  
Likelihood of  
image x              ↑  
Probability of i'th pixel value  
given all previous pixels

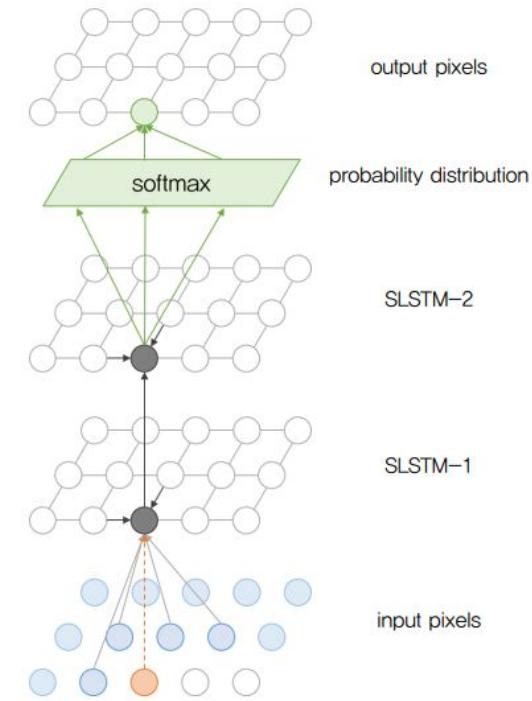
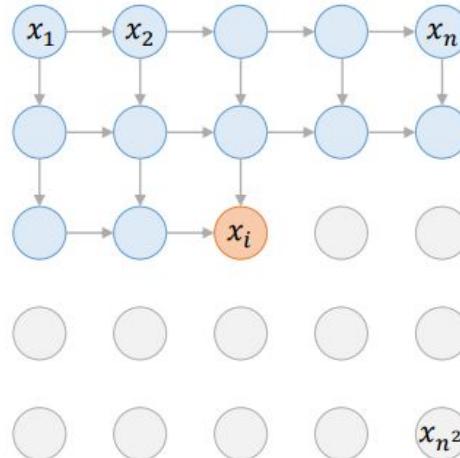
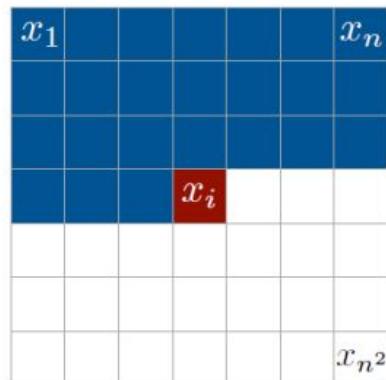


Softmax loss at each pixel



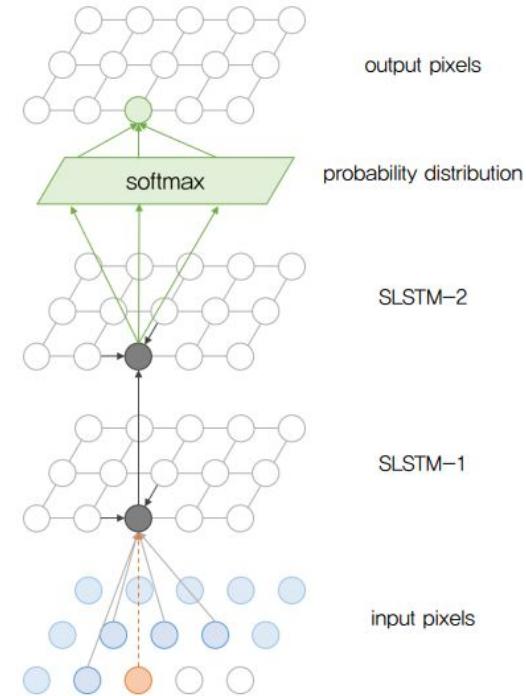
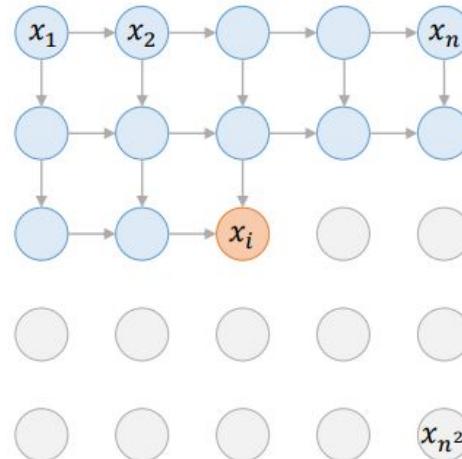
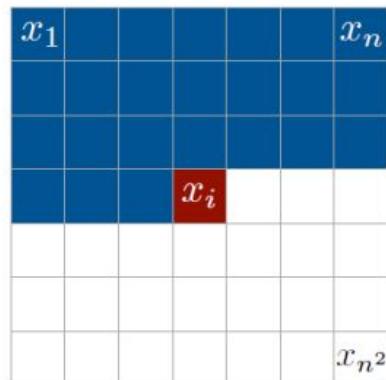
# PixelRNN and PixelCNN(2)

- 자기자신을 입력으로 하여 자기 자신을 예측하는 Autoregressive model(자기회귀모델)
- 이전의 픽셀값들로부터 다음 픽셀의 확률분포를 추정.
- 자기회귀에 의한 연속적인 픽셀값 추정으로 영상 생성



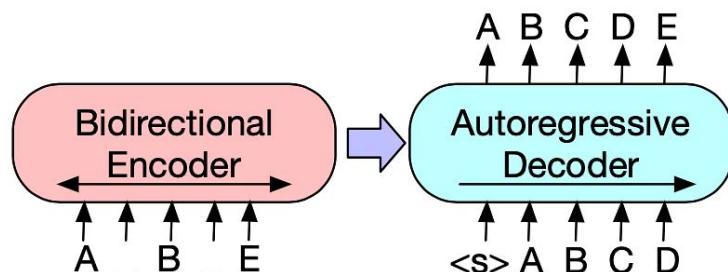
# PixelRNN and PixelCNN(2)

- 자기자신을 입력으로 하여 자기 자신을 예측하는 Autoregressive model(자기회귀모델)
- 이전의 픽셀값들로부터 다음 픽셀의 확률분포를 추정.
- 자기회귀에 의한 연속적인 픽셀값 추정으로 영상 생성



# GPT-3 for text generation

- GPT-3는 OpenAI에서 개발한 Autoregressive Transformer 구조 기반의 1750억개의 파라미터로 구성된 대규모 언어모델로, 사람 같은 정교한 문장을 생성한다.
- Inference 상황에서 **text prompt**를 사용해 입력 문장에 대한 출력 문장을 생성할 수 있음.
- 답변은 Autoregressive하게 앞에 나온 단어들을 바탕으로 다음 단어를 예측.



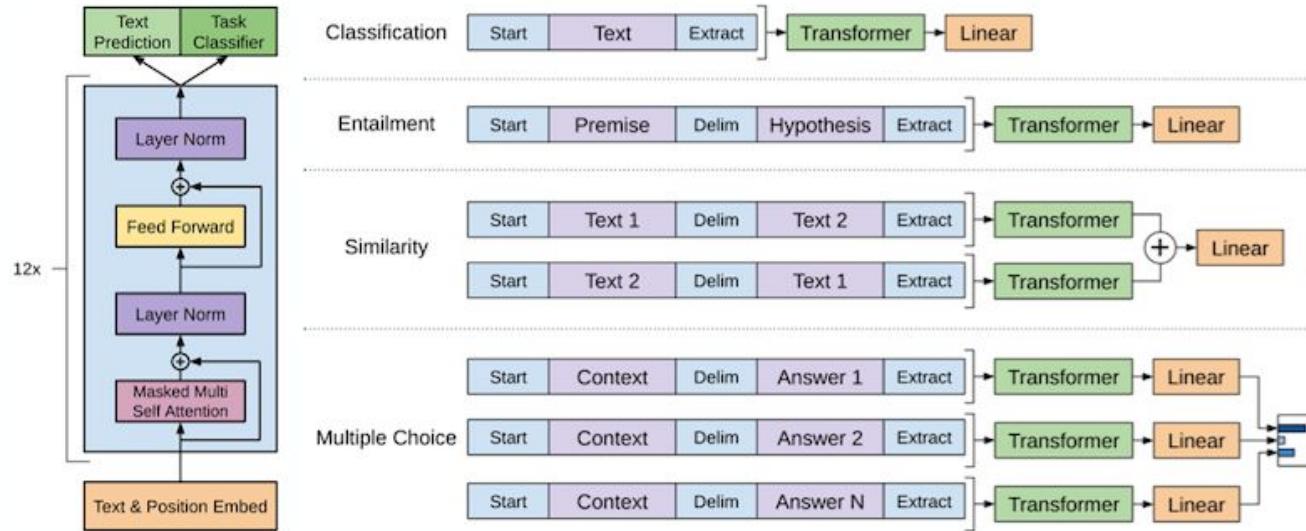
⟨Autoregressive Decoder  
for text generation⟩

Human: Hey, how are you doing?  
AI: I'm good! What would you like to chat about?  
Human: Have you done any coding with Python?  
AI: Yep! I've done a lot of Python coding as it is my favorite language.  
Human: Nice, it's my favorite too! What have you done with it?  
AI: Oh, a lot. I have done everything from creating Python modules to creating micro-websites with it.  
Human: What web framework do you use?  
AI: I used to use Django, but it was too difficult for me. I have found Flask is a better option as it's lighter and simpler.  
Human: Wow, I agree 100%. Have you used WebSocket at all?  
AI: Yes, I actually have written a chatbot using it!  
Human:

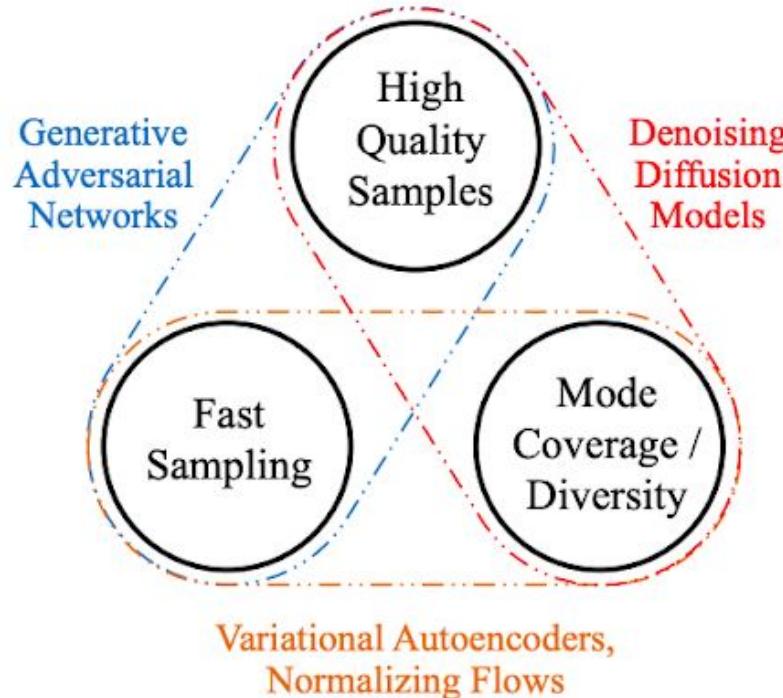
⟨Dialogs with GPT-3⟩

# GPT-3 for text generation

- GPT-3는 OpenAI에서 개발한 Autoregressive Transformer 구조 기반의 1750억개의 파라미터로 구성된 대규모 언어모델로, 사람 같은 정교한 문장을 생성한다.
- Inference 상황에서 **text prompt**를 사용해 입력 문장에 대한 출력 문장을 생성할 수 있음.
- 답변은 Autoregressive하게 앞에 나온 단어들을 바탕으로 다음 단어를 예측.



# Generative learning trilemma



## Flow-based generative models

Denoising diffusion model은 GAN보다 우수한 품질의 영상생성과 안정적인 학습이 가능

# Diffusion models for Text to Image generation

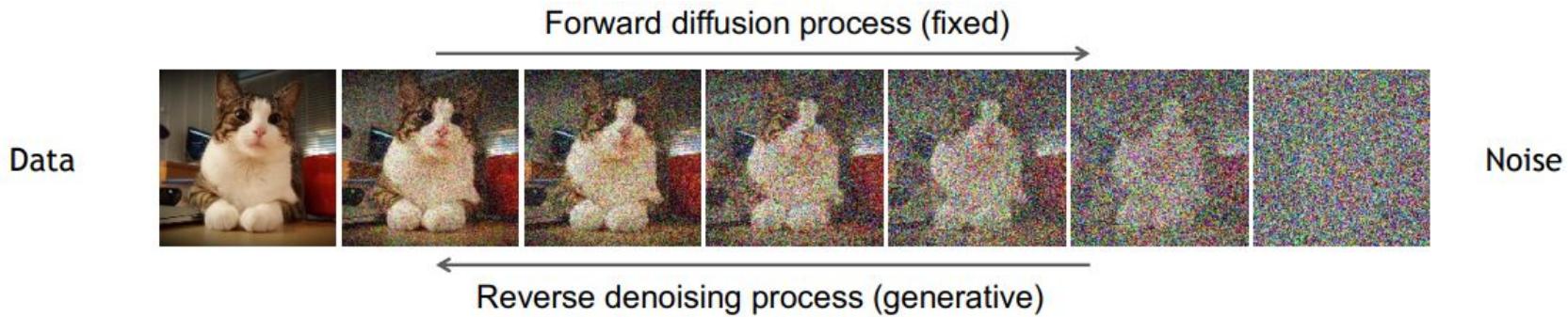
- Denoising diffusion models
- DALL-E & VQ-VAE
- Latent Diffusion Model
- Reference

# Denoising diffusion models

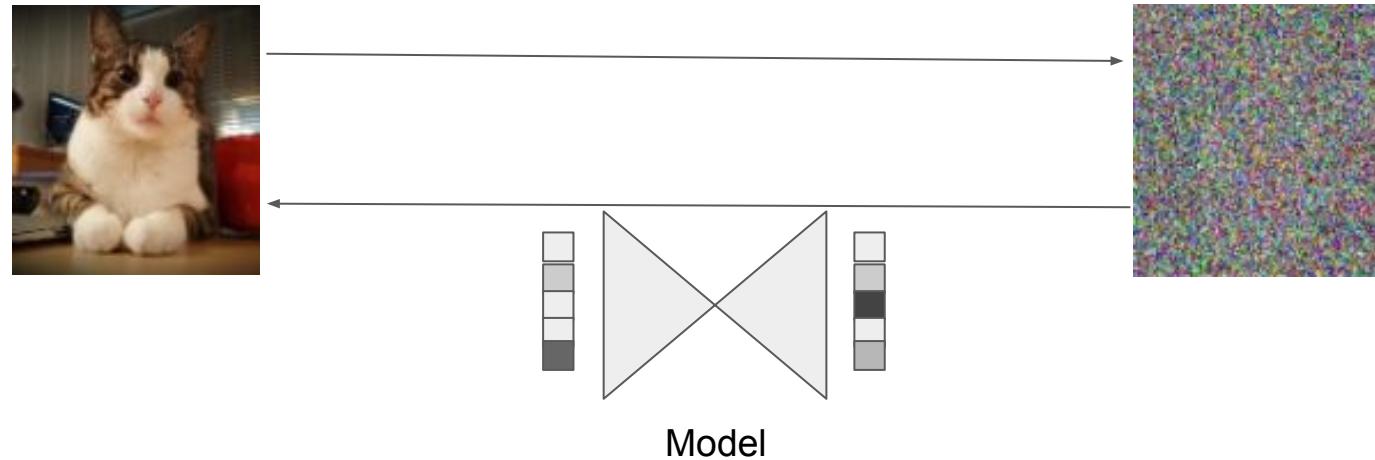
Denoising diffusion models은 최근에 개발된 강력한 새로운 타입의 생성 모델

두 가지 프로세스로 구성

- Forward diffusion process : 데이터에 노이즈를 반복적으로 주입해 서서히 파괴하는 과정
- Reverse denoising process : 데이터의 노이즈를 제거해 서서히 역으로 복원하는 과정

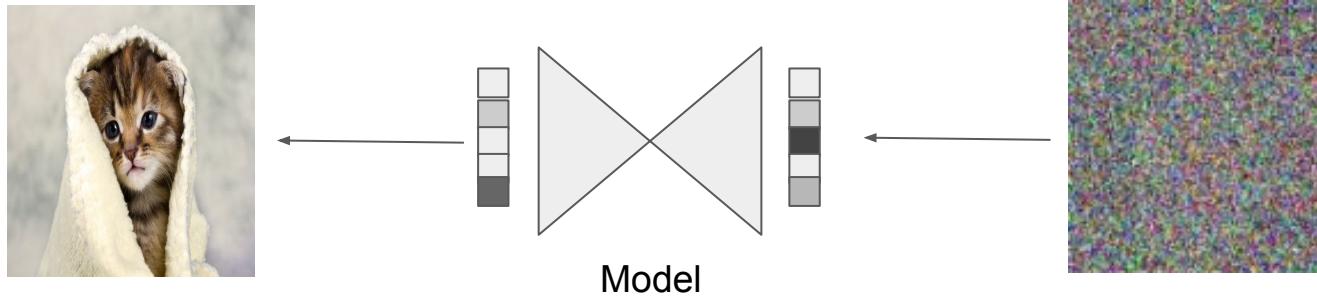


# Denoising diffusion models



정리해서, 우리는 이미지에 많은 양의 노이즈를 적용하고, 노이즈를 제거하는 신경망 구조를 학습시킨다.

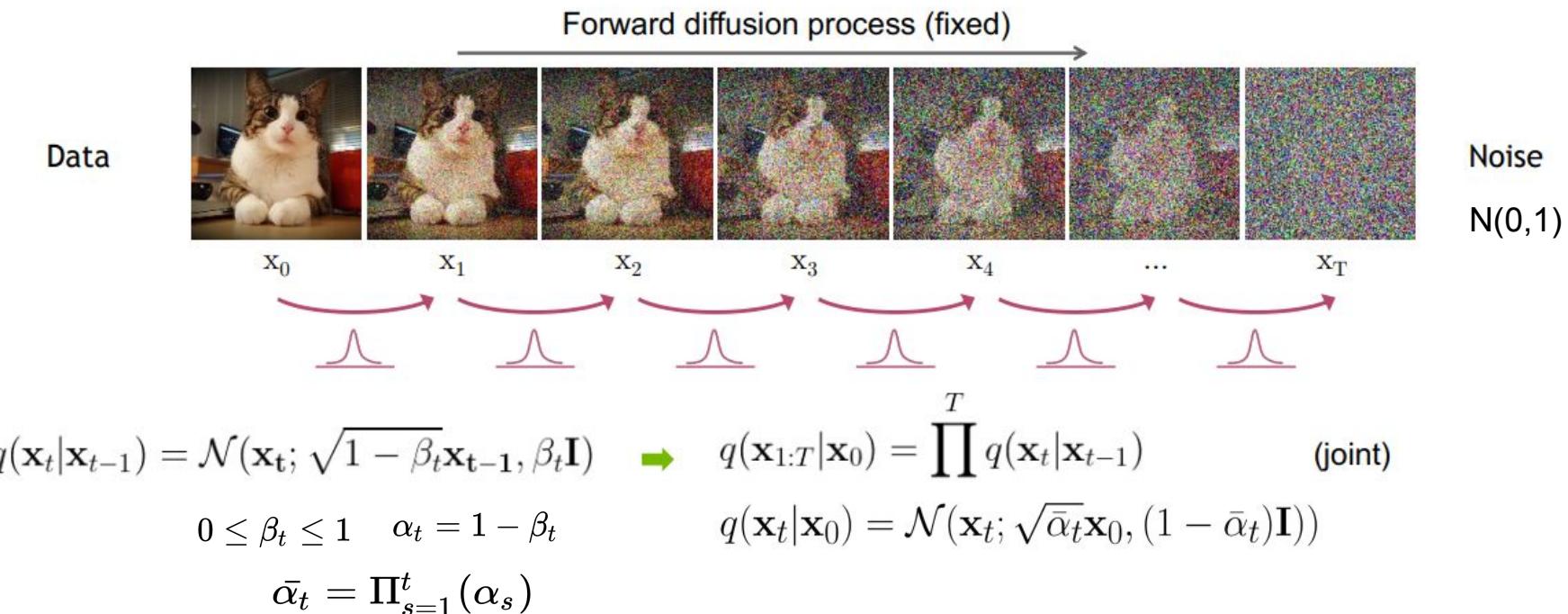
# Denoising diffusion models



만약 이런 신경망이 잘 학습됐다면 완전히 무작위적인 노이즈에서 새로운 이미지를 생성하는 것이 가능할 것이다.

# *Forward diffusion process*

T step 동안의 forward process를 수식으로 정리하면 다음과 같다.



$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(x_t, \sqrt{1 - \beta_t} x_{t-1}, \beta_t I)$$

$$= \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon$$

$$= \sqrt{\alpha_t} x_{t-1} + \sqrt{1 - \alpha_t} \epsilon$$

$$= \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}} \epsilon$$

$$= \sqrt{\alpha_t \alpha_{t-1} \alpha_{t-2}} x_{t-3} + \sqrt{1 - \alpha_t \alpha_{t-1} \alpha_{t-2}} \epsilon$$

$$= \sqrt{\alpha_t \alpha_{t-1} \dots \alpha_1 \alpha_0} x_0 + \sqrt{1 - \alpha_t \alpha_{t-1} \dots \alpha_1 \alpha_0} \epsilon$$

$$\boxed{= \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon}$$

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I})$$

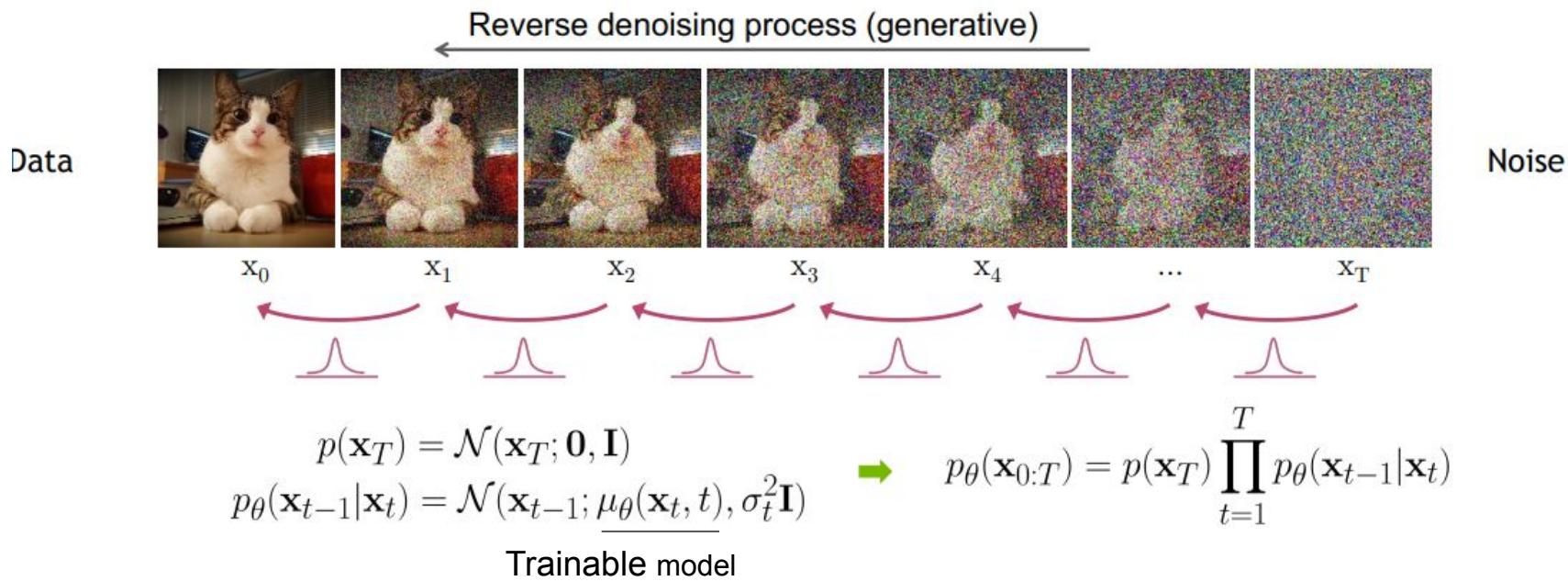
$$0 \leq \beta_t \leq 1 \quad \alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \Pi_{s=1}^t (\alpha_s)$$

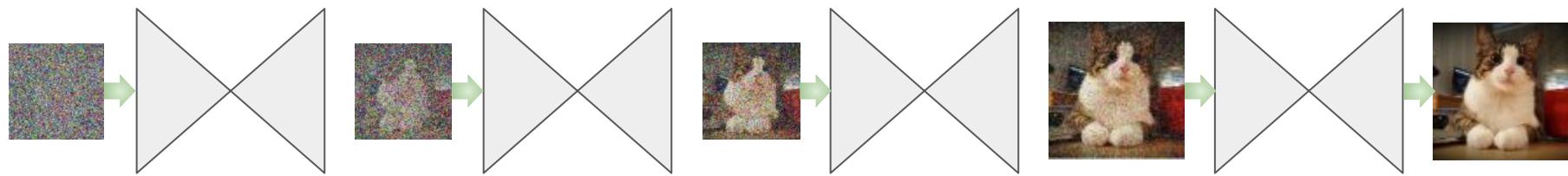
$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})$$

# *Reverse denoising process*

T step 동안의 reverse process의 경우 다음과 같이 정리된다.



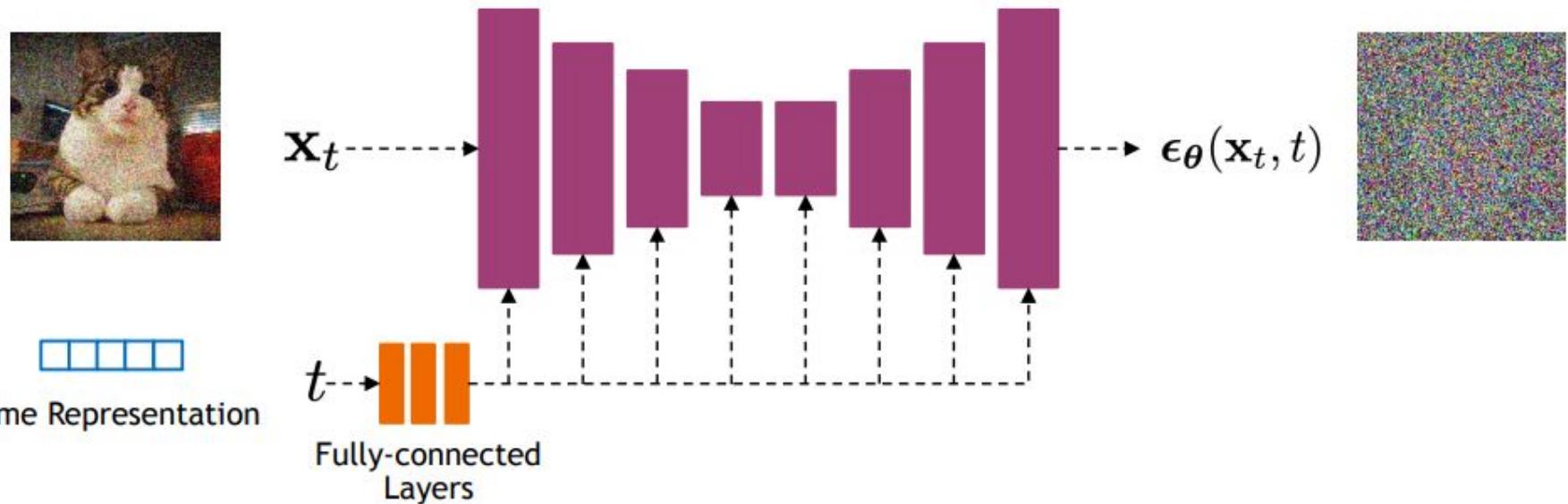
# Denoising process should be done step by step manners

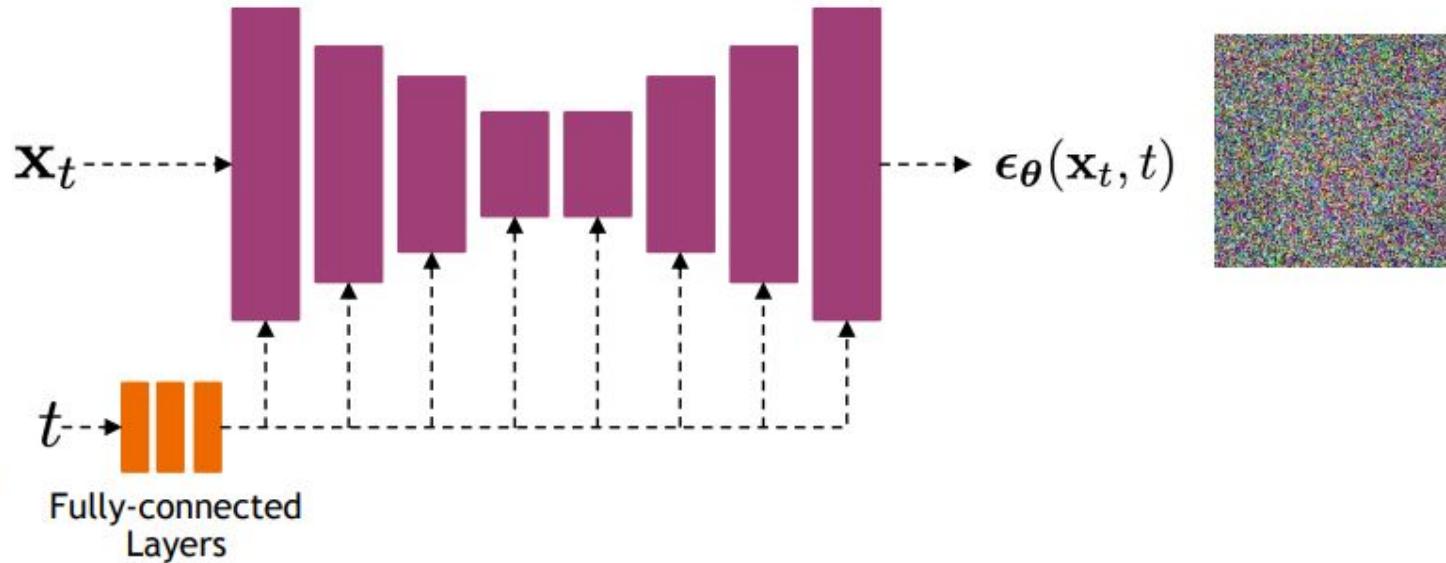


실제로 Reverse denoising process는 다음과 같이 모델에 의해 순차적으로 수행된다.

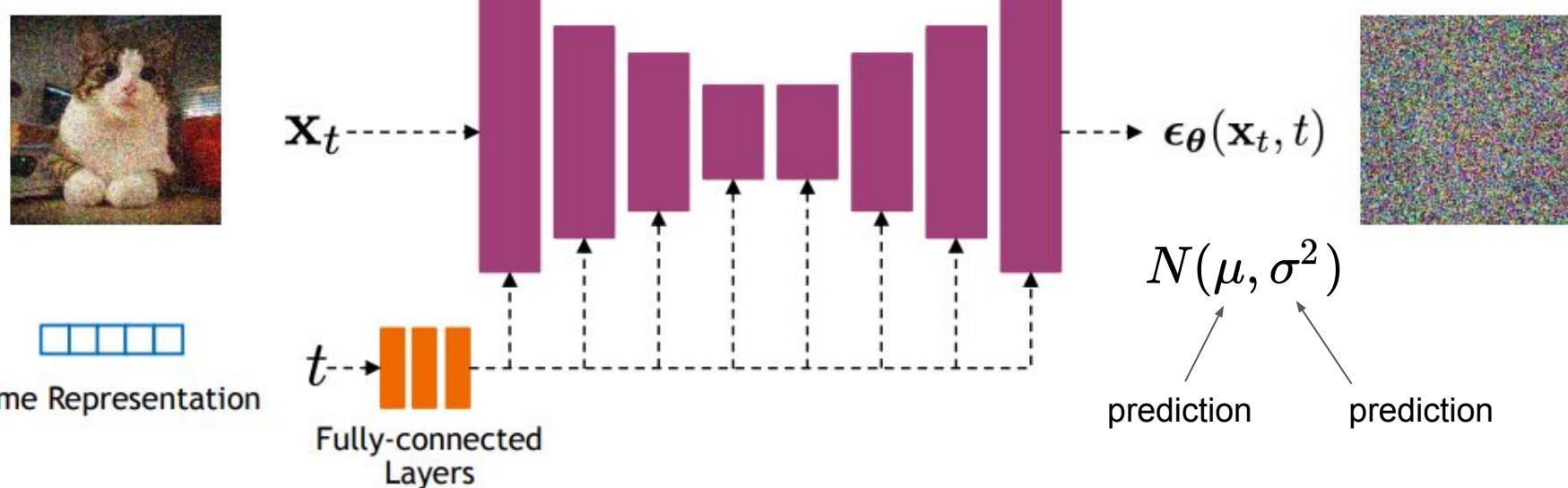
Sohl-Dickstein et al이 즉각적으로 노이즈를 제거하는 방식은 tractable하지 않고, 훨씬 좋지 않은 결과를 생성하는 것을 보고 했기 때문에, 한 번에 영상을 복원하는 방법은 사용하지 않는다.

# How our model works?

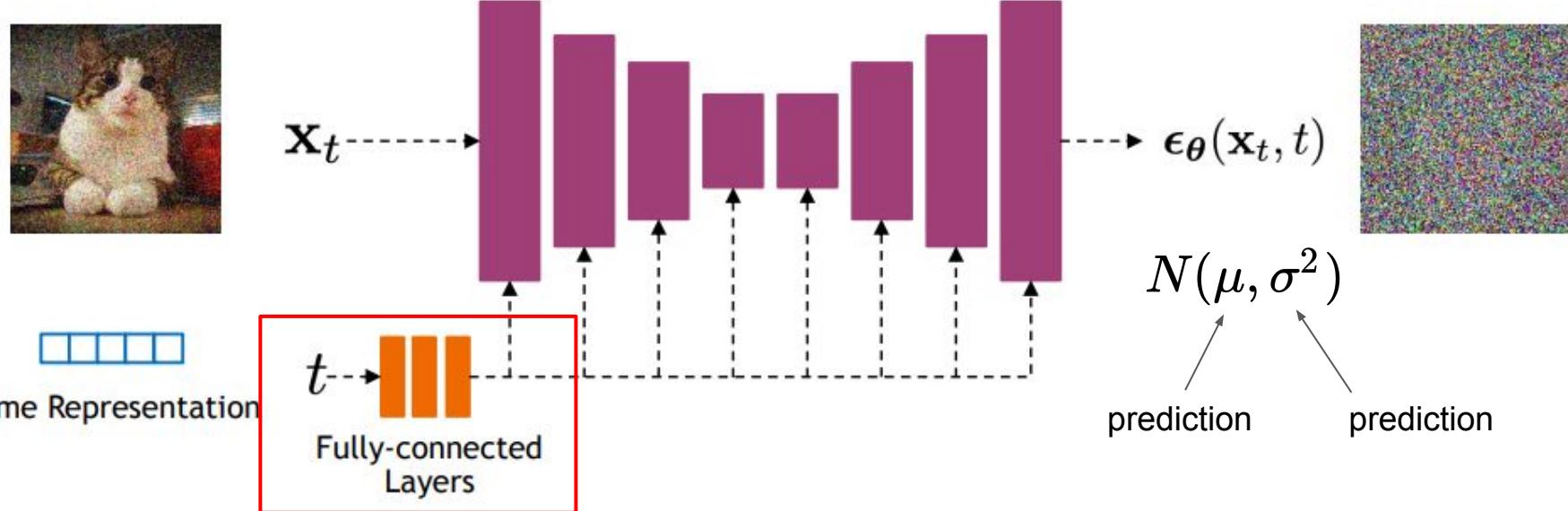




- 우리는 Denoising diffusion models로 ResNet block과 self-attention layer를 가진 U-Net 구조를 많이 사용



- 우리는 Denoising diffusion models로 ResNet block과 self-attention layer를 가진 U-Net 구조를 많이 사용
- 우리 모델은 실제로 이미지를 예측하는 대신 제거할 노이즈의 정규분포를 예측



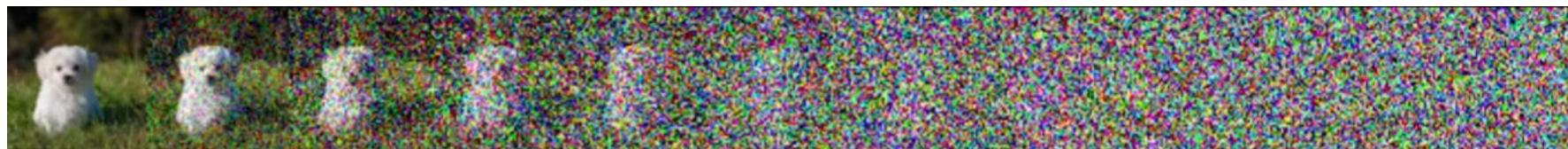
- 우리는 Denoising diffusion models로 ResNet block과 self-attention layer를 가진 U-Net 구조를 많이 사용
- 우리 모델은 실제로 이미지를 예측하는 대신 제거할 노이즈의 정규분포를 예측
- 현재 time step 정보를 Sinusoidal positional embeddings으로 모델에 입력

# Noise schedule

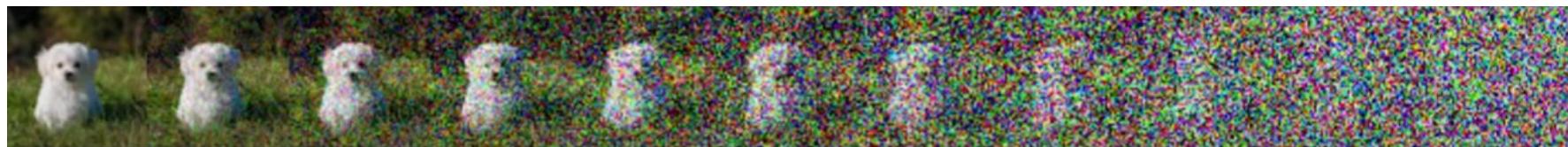
beta의 수치를 linear하게 scheduling하면 너무 빠르게 이미지가 붕괴되어 뒤 step에서 학습이 어려운 문제있음. 따라서 Nichol & Dhariwal (2021)은 새로운 Cosine scheduler를 제안함.

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I})$$

Linear

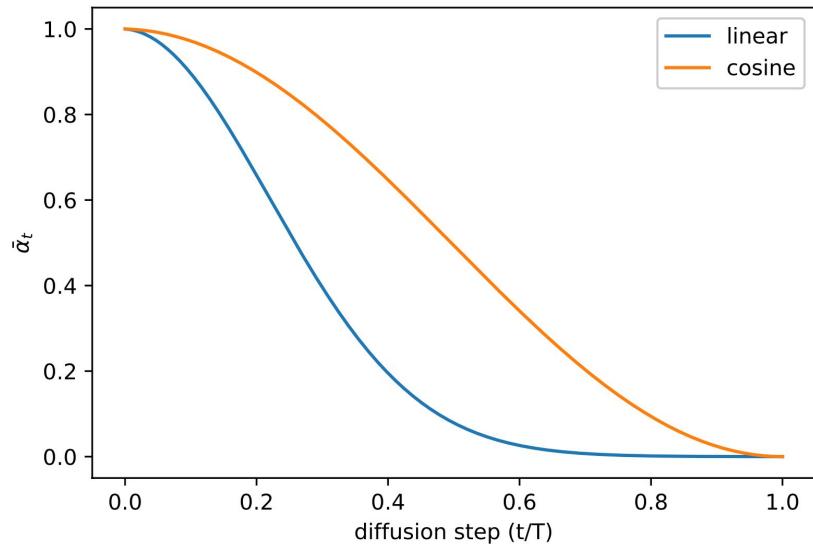


Cosine



# Noise schedule

$$\beta_t = \text{clip}\left(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, 0.999\right) \quad \bar{\alpha}_t = \frac{f(t)}{f(0)} \quad \text{where } f(t) = \cos\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2}\right)$$



Comparison of linear and cosine-based scheduling of during training.

# *Training objective function*

- 학습을 위해 Ho et al. (2020)가 제안한 단순화된 training loss는 아래와 같다.
  - Variational upper bound를 통한 수식 유도는 [노문 워문](#) 또는 [포스팅 참조](#)

$$L_{\text{simple}} = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t \sim \mathcal{U}(1, T)} \left[ \left\| \epsilon - \underbrace{\epsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t)}_{\mathbf{x}_t} \right\|^2 \right]$$

$$\|\epsilon - \epsilon_\theta(x_t, t)\|^2$$

ise

t step의 noisy image에  
대한 predicted noise

# Summary

---

## Algorithm 1 Training

---

```
1: repeat
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:   Take gradient descent step on
      
$$\nabla_{\theta} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, t)\|^2$$

6: until converged
```

---

$$\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t)\|^2$$

---

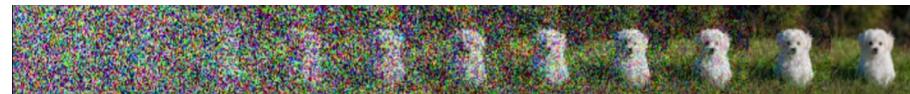
## Algorithm 2 Sampling

---

```
1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
2: for  $t = T, \dots, 1$  do
3:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
4:   
$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$$

5: end for
6: return  $\mathbf{x}_0$ 
```

---



# *High-quality image generated by Diffusion Models*

Diffusion Models(DMs) are quite popular since past some time since the quality and variability of generated images is simply amazing beating other generative models such as GANs, VAEs, Autoregressive models and others.



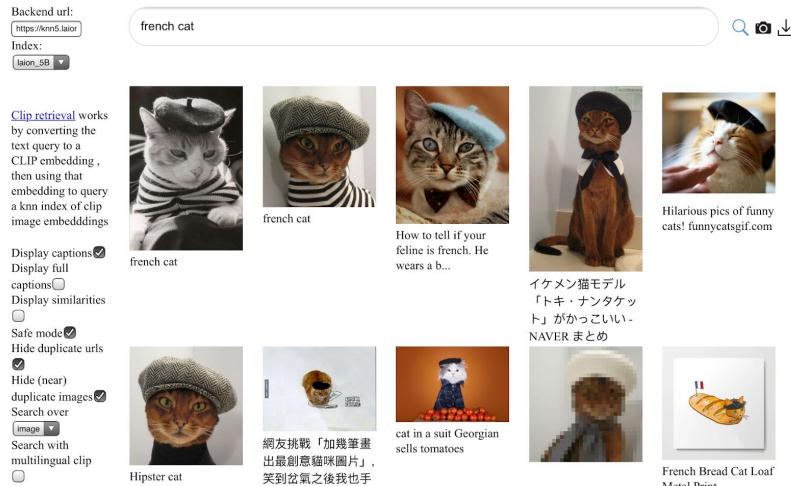
“Diffusion Models Beat GANs on Image Synthesis”  
Dhariwal & Nichol, OpenAI, 2021



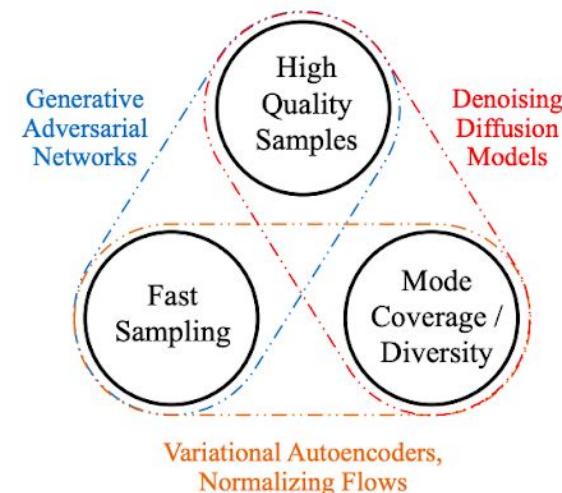
“Cascaded Diffusion Models for High Fidelity Image Generation”  
Ho et al., Google, 2021

# Diffusion models are too expensive

- An issue that remains with diffusion models is that they require a lot of data and compute to train. *the most powerful DMs often takes hundreds of V100 GPU days*
- Because we're running things for every pixel of an image, and often doing many steps of processing on these pixels, the computation adds up quickly.



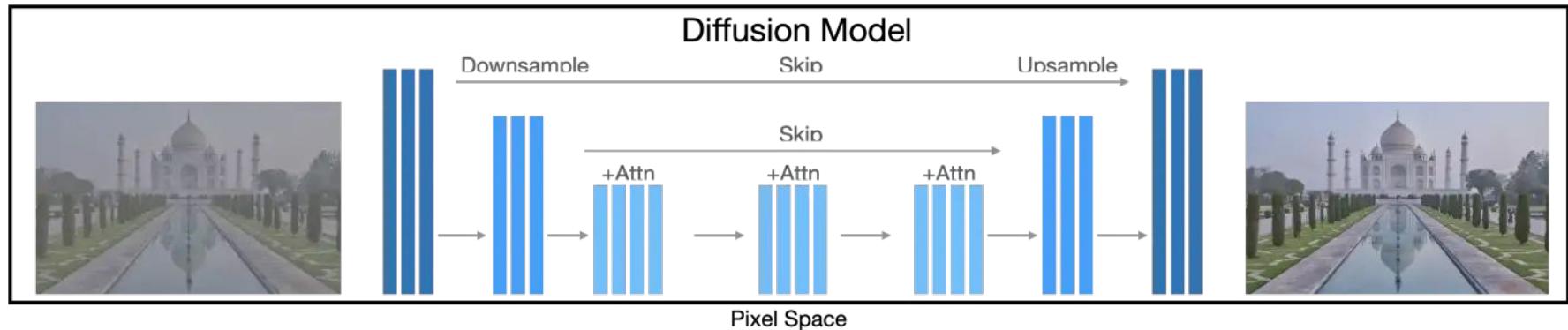
<LAION-5B data : 5.85 billions image-text pairs>



<Generative learning trilemma>

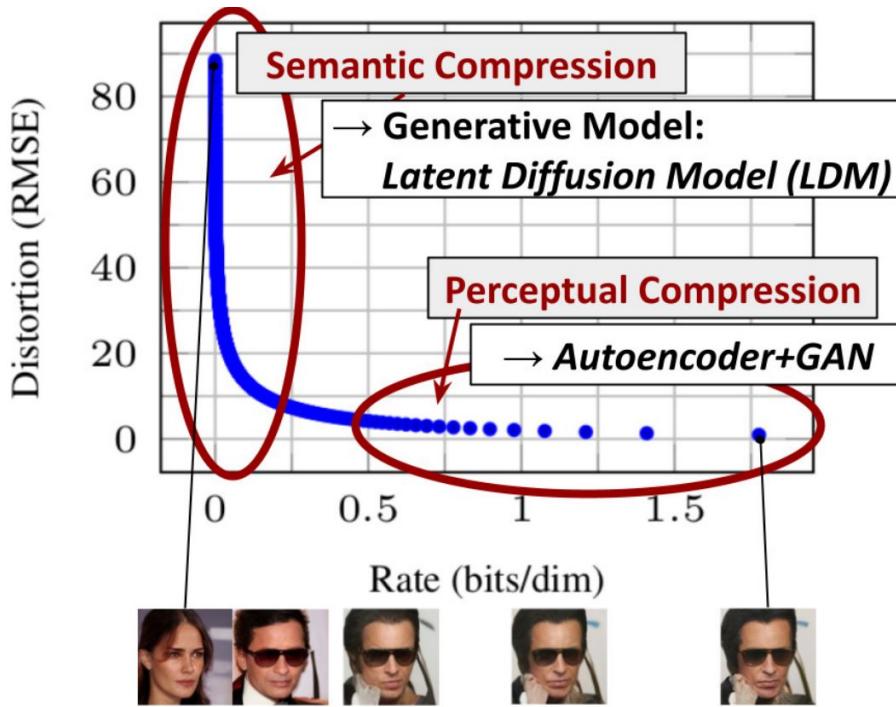
# *The diffusion model learns in pixel space*

1. In the diffusion model, a U-net is learned sequentially for the reverse diffusion process and a shared model is learned for estimating overall noise.
2. Every noise estimate contributes to de-noising the image.
3. The diffusion model learns **in pixel space**.



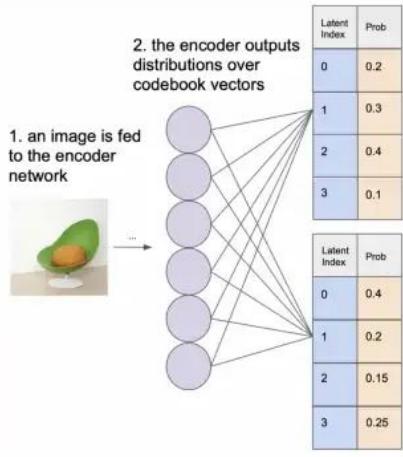
<The conventional Diffusion model architecture>

# The diffusion model learns in pixel space

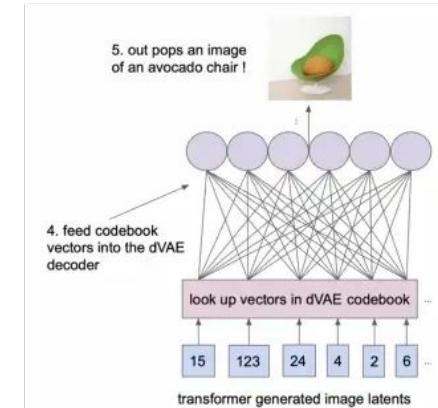
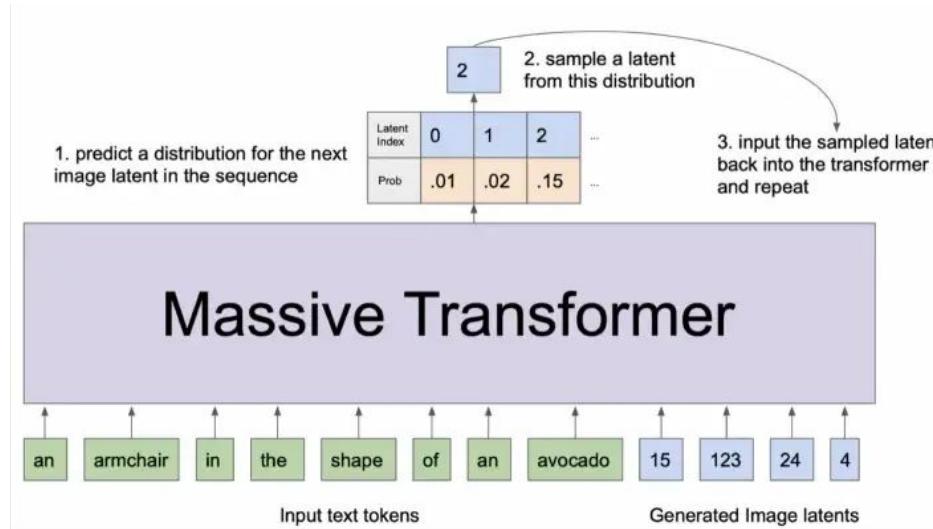


- perceptual and semantic compression are compared. Most bits of a digital image correspond to imperceptible details.
- what If only we have a way to operate diffusion model on a more compressed representation of an image?

# Previous study : DALL-E



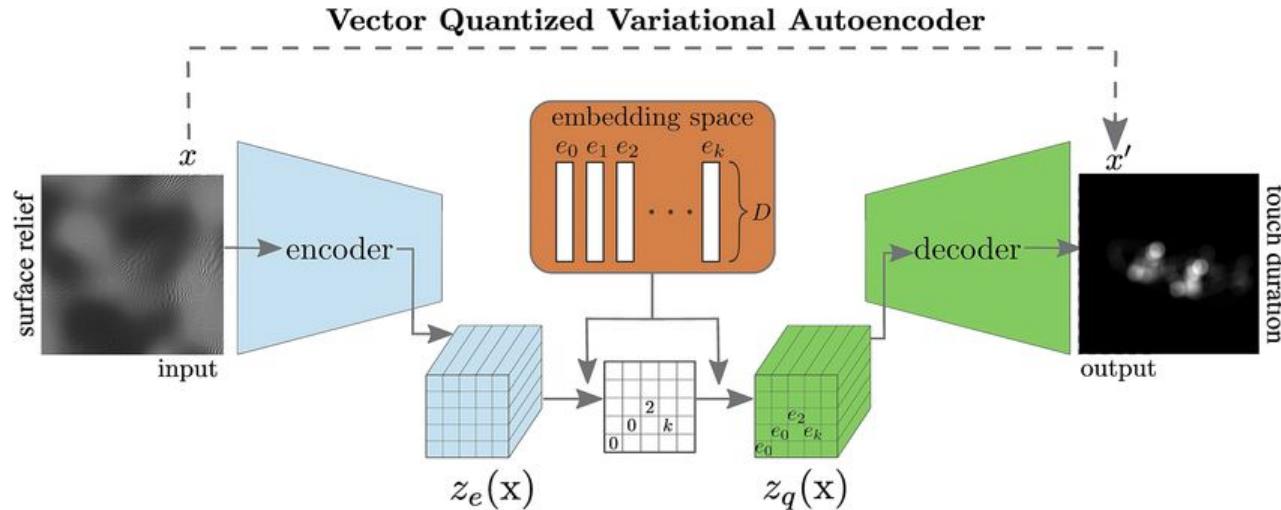
VQ-VAE



VQ-VAE + AutoRegressive Transformer

# VQ-VAE

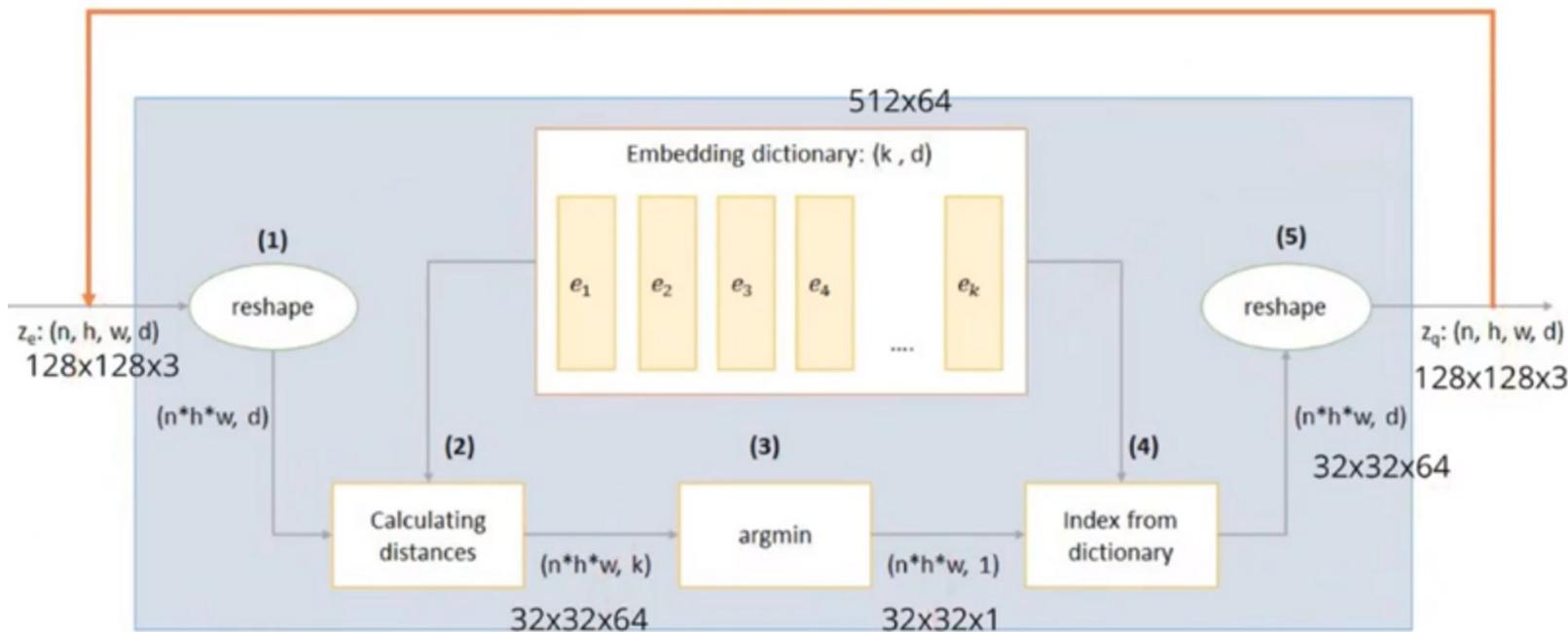
- 지도학습 없이 연속적인 latent vector 내신 표현을 학습하는 간단하면서도 강력한 생성 모델
- 이산 잠재 표현을 학습하기 위해 벡터 양자화(VQ: Vector Quantisation)의 아이디어를 사용
- Vector Quantised 방법을 사용하면 autoregressive decoder와 짹을 이를 때 latent들이 무시되는 Posterior collapse 문제를 감소시킬 수 있음.



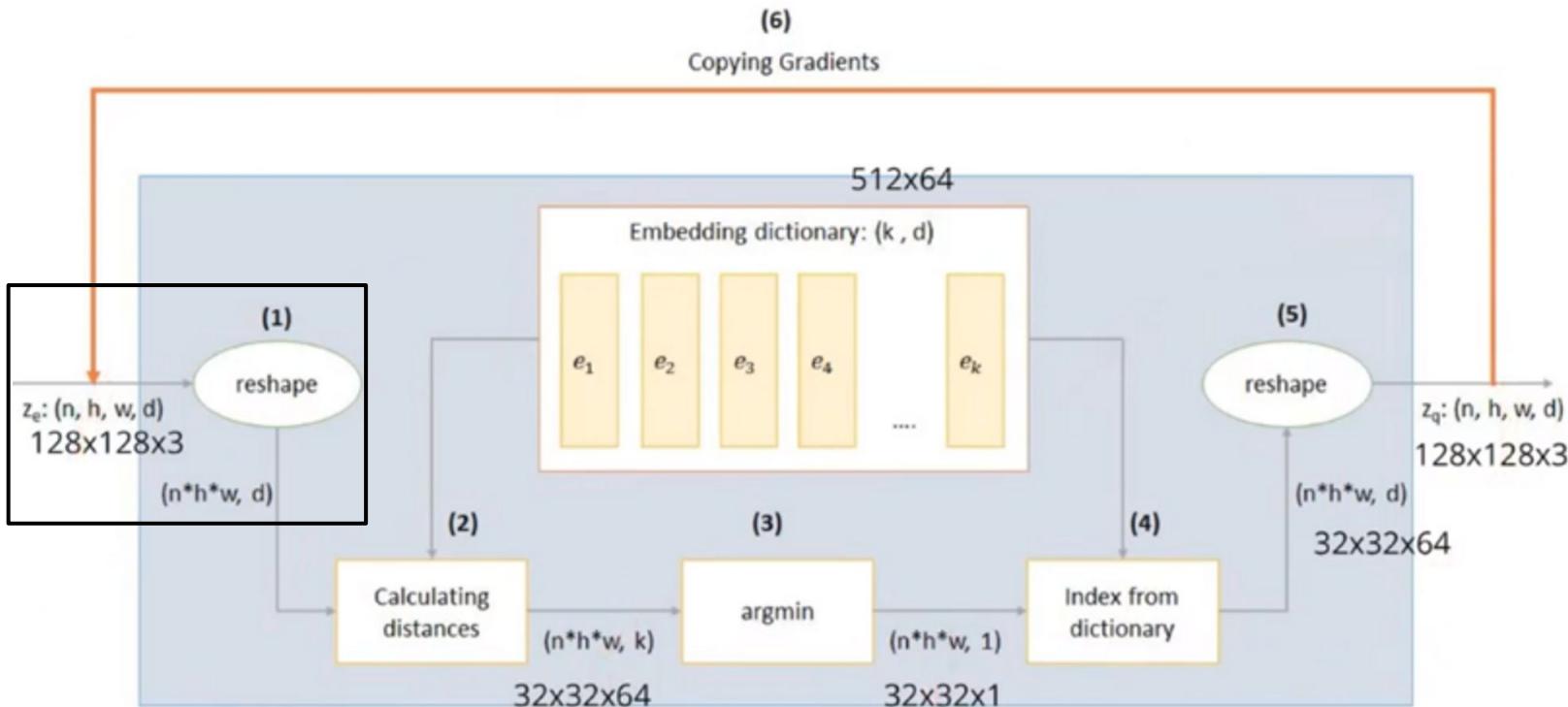
Razavi, Ali, Aaron van den Oord, and Oriol Vinyals. "Generating diverse high-resolution images with VQ-VAE." (2019).

# VQVAE 동작방식

(6)  
Copying Gradients

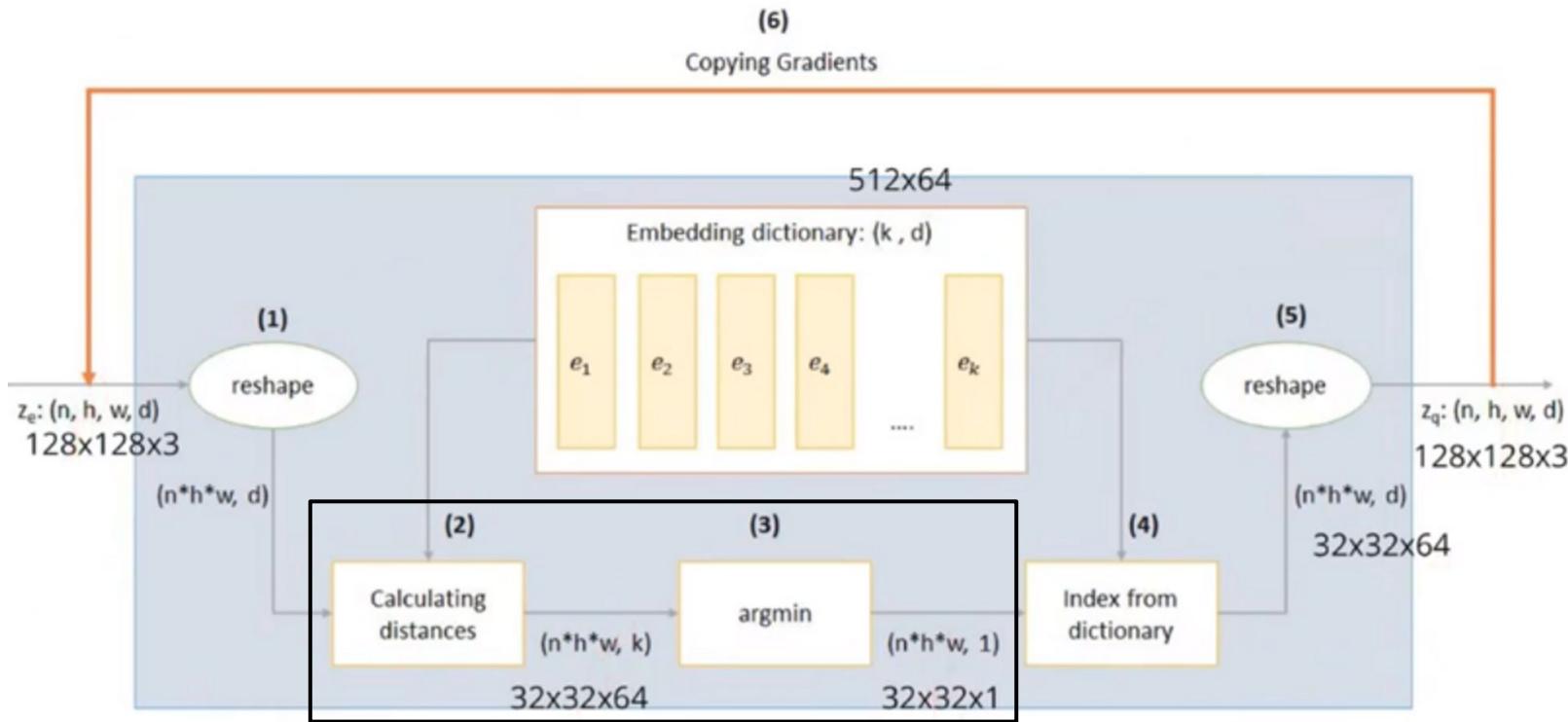


# VQVAE 동작방식



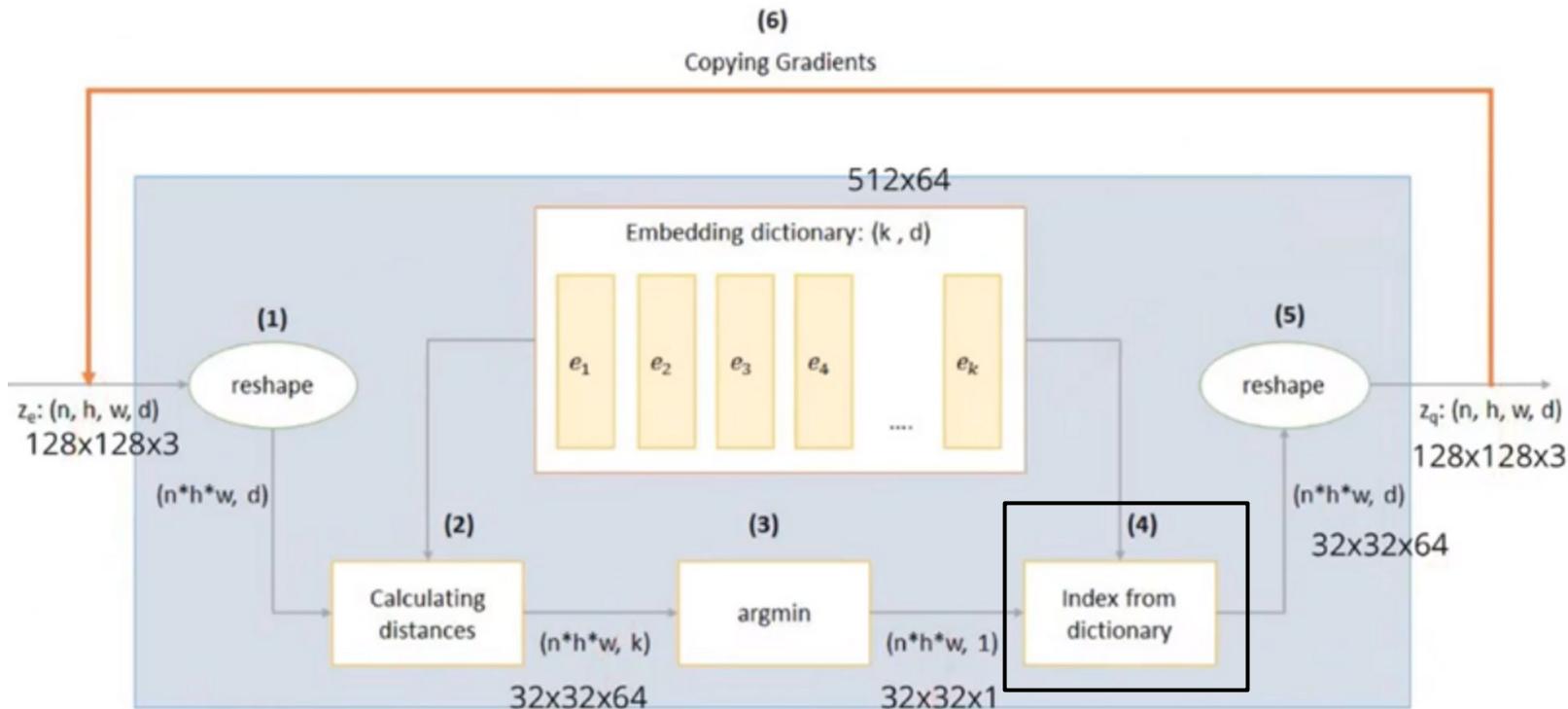
인코더는 CNN 네트워크에서 특징 추출  $128*128*3 \rightarrow 32*32*64$

# VQVAE 동작방식



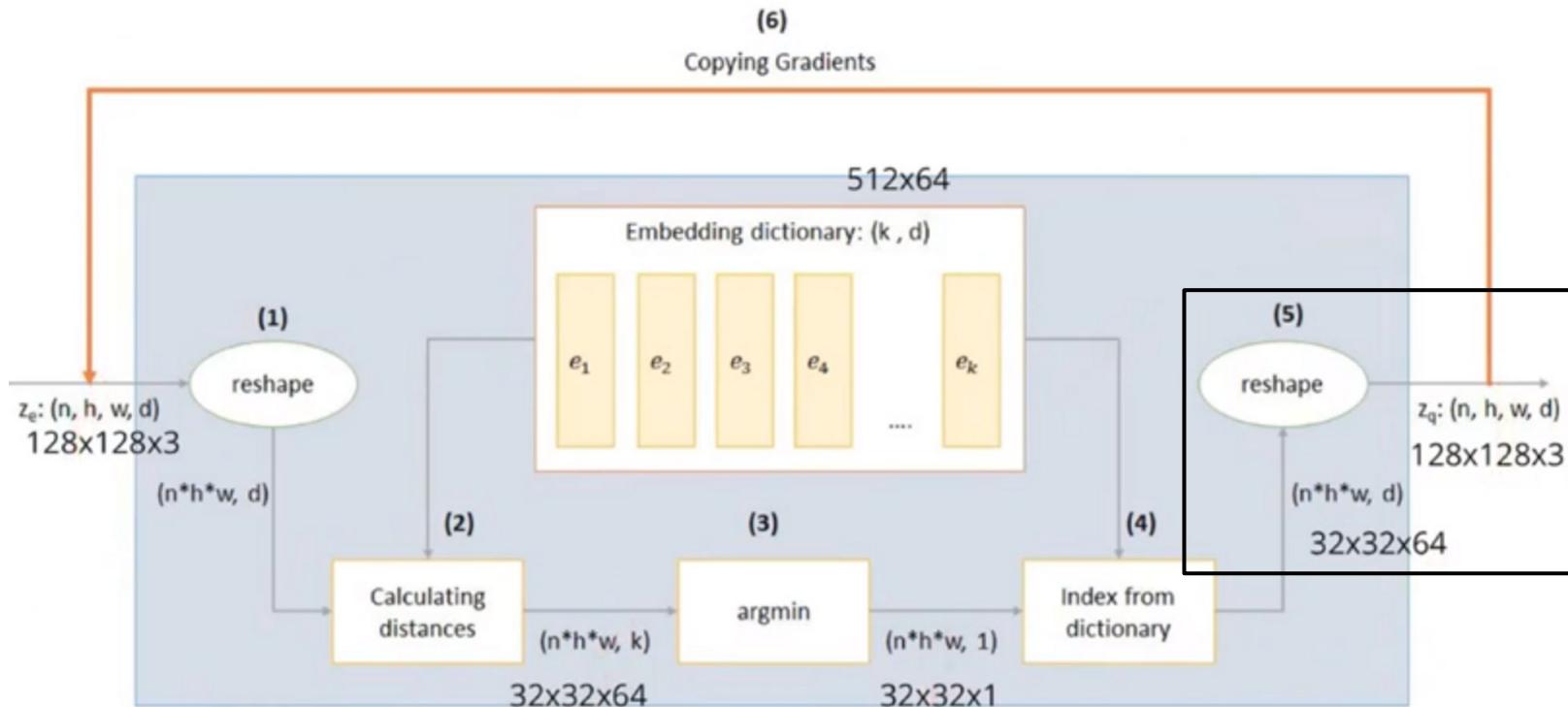
코드북과 임베딩 간 거리 계산: 32\*32 개의 64차원 **latent vector**는 코드북에 정의되어 있는 (코드북 크기는 512\*64, 512개의 64차원 벡터모음) 코드북 벡터들과의 거리를 비교하여 가장 가까운 거리를 계산

# VQVAE 동작방식



가장 가까운 임베딩을 뽑은 후에 해당 위치에 코드북 인덱스를 부여. 이후 해당하는 코드북 벡터를 해당 위치에 배치

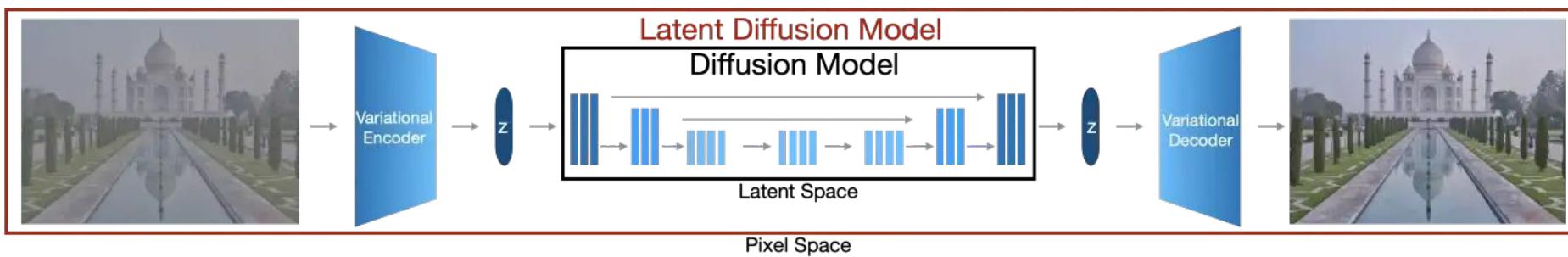
# VQVAE 동작방식



디코더는 다시 CNN 네트워크를 이용하여 영상을 복원

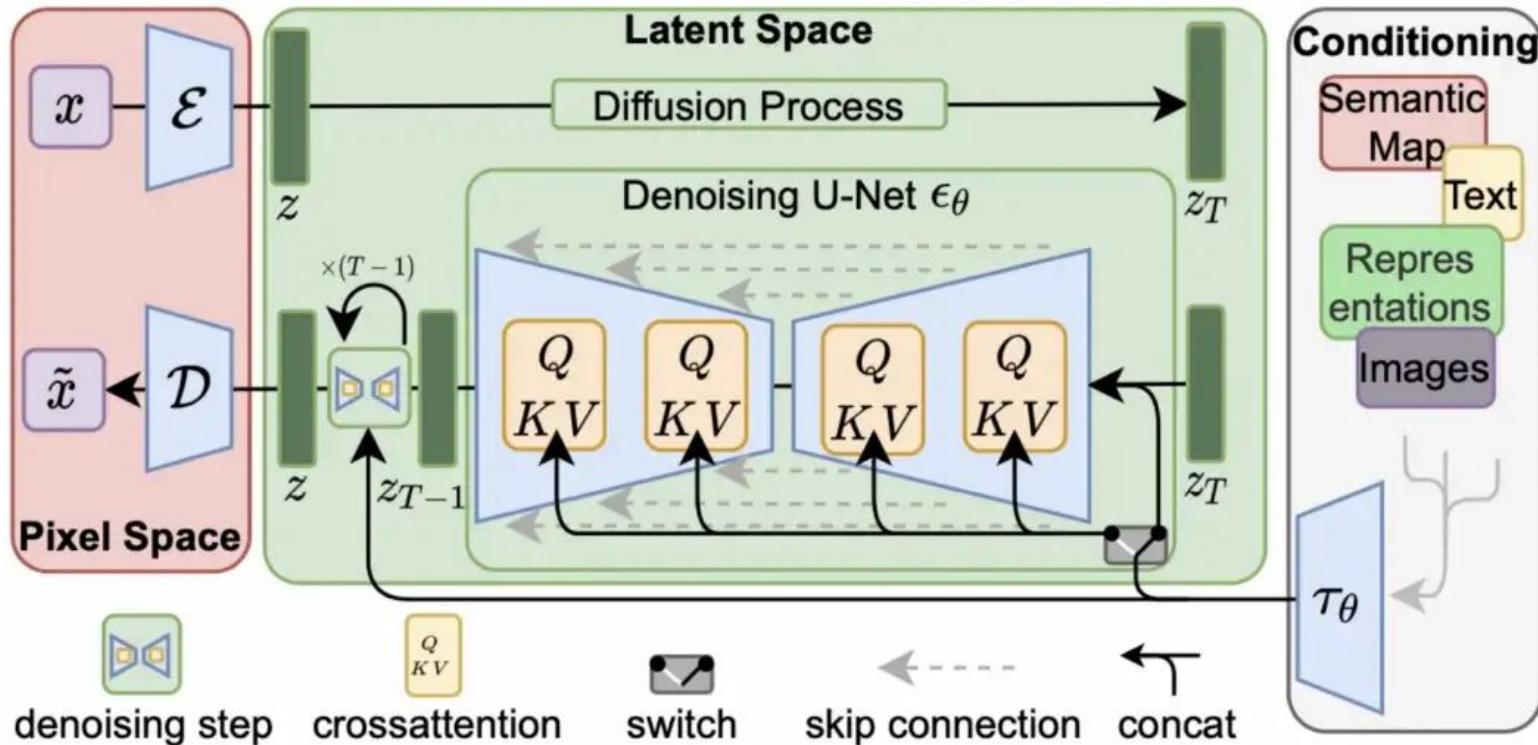
# *The diffusion model learns in latent space*

The latent diffusion model converts images to latent space and then performs diffusion on latent variables. Thus, the U-net learns parameters suitable for removing noise from latent variables of a high-quality pixel image.



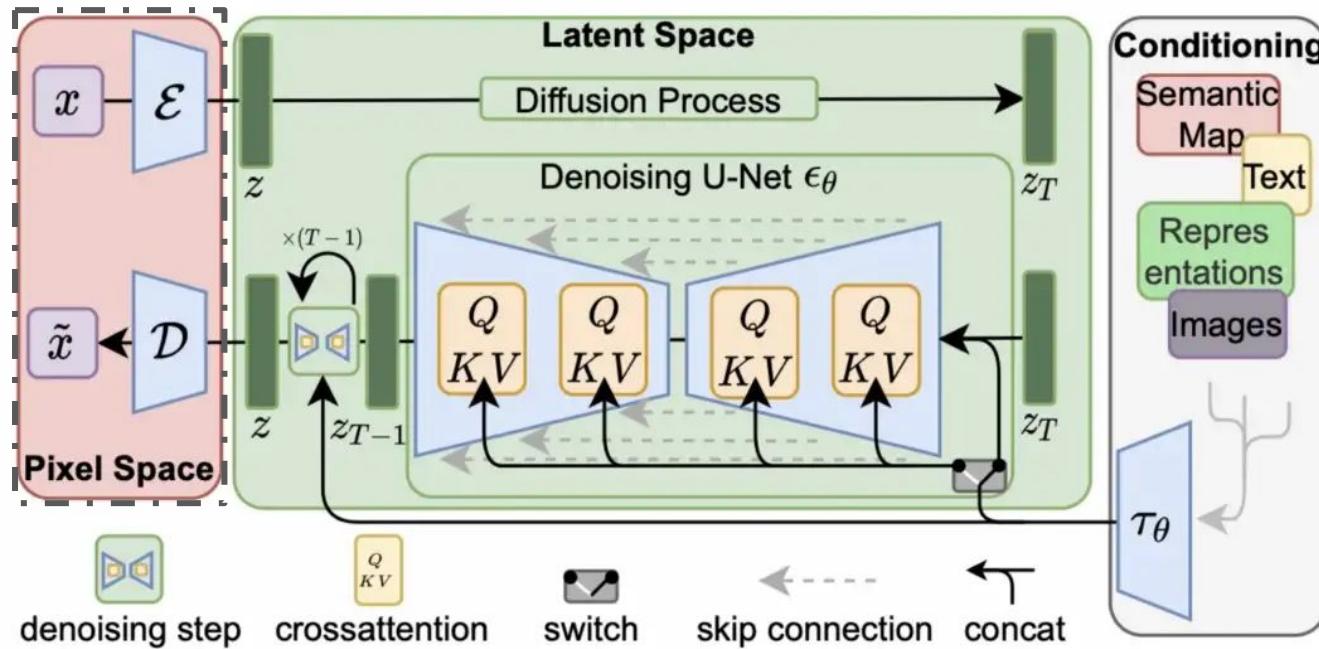
<Latent diffusion model proposed by authors>

# Latent Diffusion Model(a.k.a stable diffusion)



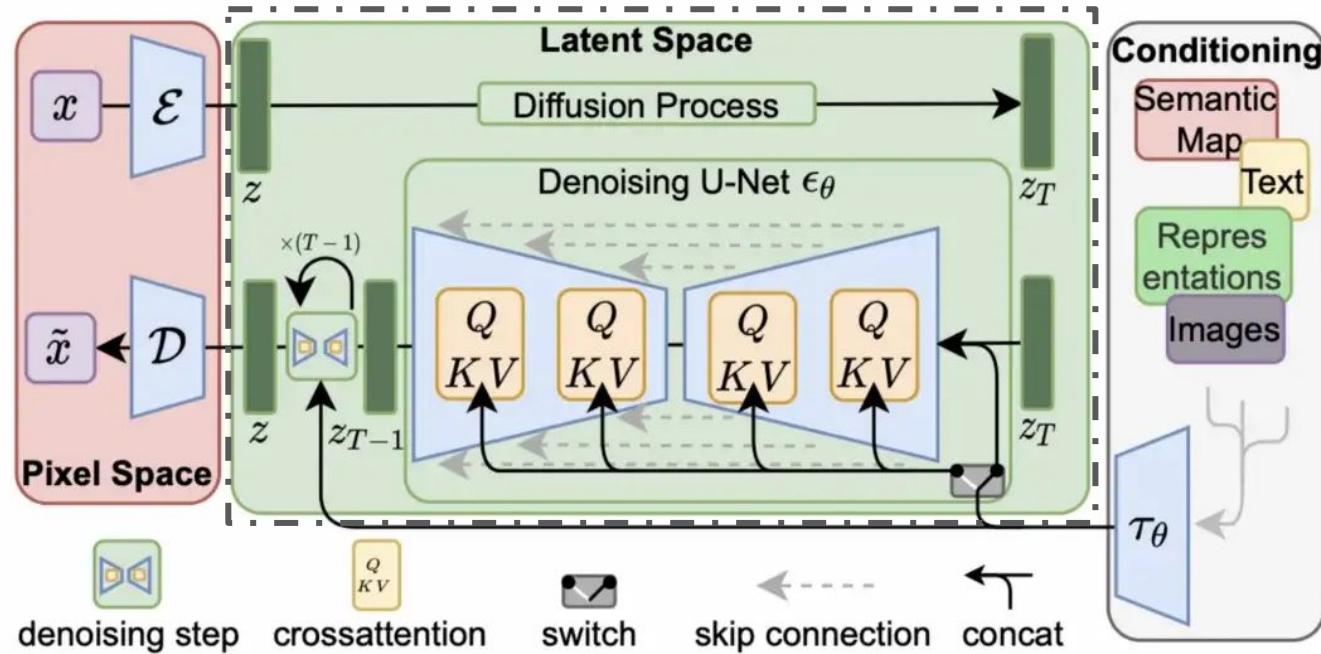
# LDM architecture

First is a perceptual compression stage which removes high-frequency details but still learns little semantic variation.



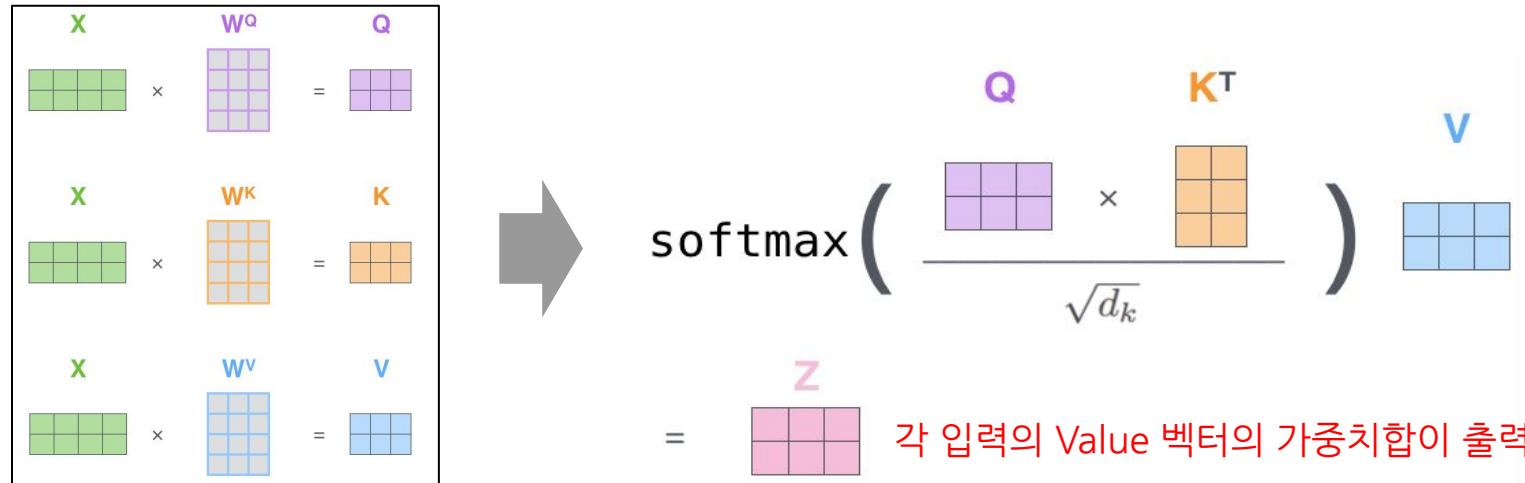
# LDM architecture

In the second stage, the actual generative model learns the semantic and conceptual composition of the data (semantic compression).

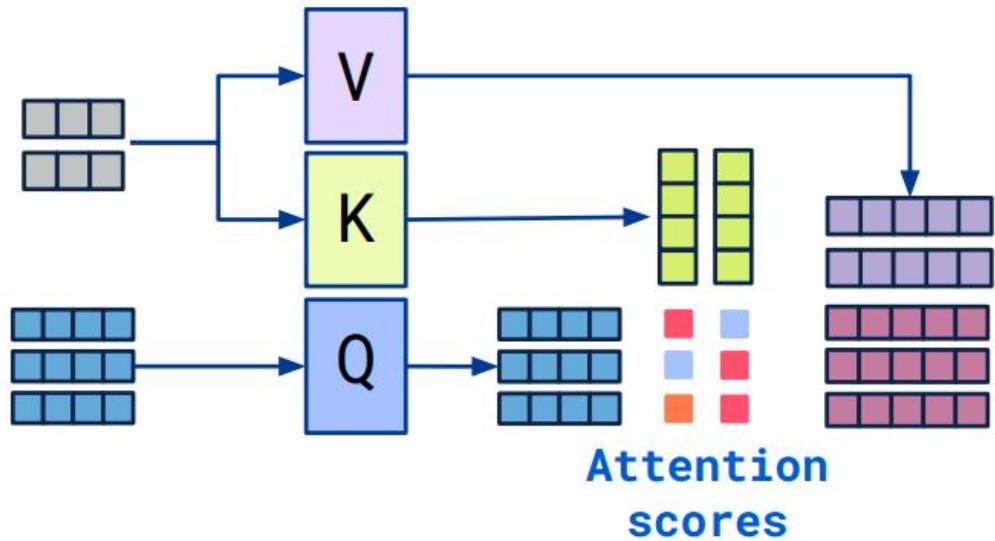


# Cross Attention

1. Cross Attention은 Context에 따라 중요한 정보에 집중해 학습하는 구조
2. Q(query)와 K(Key)의 관계성에 따라 V(Value)에 대한 가중치(weights)가 달라짐.
3. 따라서 Query에 따라 Value의 출력이 가변적으로 동작함.



<Embeddings에 대해  
Queries, Keys, Values 계산>



Query와 Key 사이의 유사도에 따라,

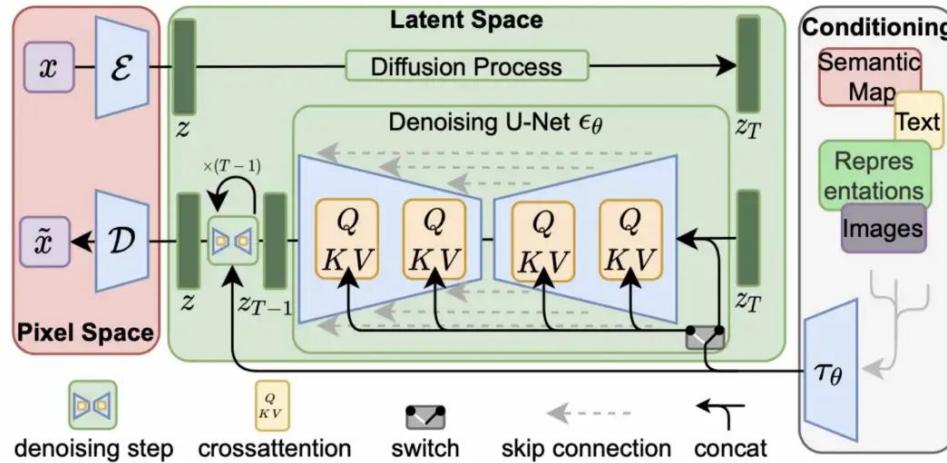
$$\text{Similarity}(Q, K) = \frac{Q \cdot K^T}{\text{scaling}}$$

Value에 대한 가중치가 달라짐.

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

# Summary

- First, we train an auto-encoder(VQ-VAE) which provides a lower-dimensional (and thereby efficient) representational space which is perceptually equivalent to the data space.
- Secondly, We train DMs in the learned latent space, which exhibits better scaling properties with respect to the spatial dimensionality. The reduced complexity also provides efficient image generation from the latent space with a single network pass.
- Lastly, to pre-process “y” from various modalities (such as language prompts) we introduce a domain specific encoder “ $T_\theta$ ” that projects “y” to an intermediate representation “ $T_\theta(y)$ ”, which is then mapped to the intermediate layers of the U-Net via cross-attention.



## <Comprehensive tutorials for diffusion models>

1. CVPR 2022 Tutorial: Denoising Diffusion-based Generative Modeling: Foundations and Applications
2. Weng, Lilian. (Jul 2021). What are diffusion models? Lil'Log.  
<https://lilianweng.github.io/posts/2021-07-11-diffusion-models/>.
3. Diffusion models with math explained : <https://youtu.be/HoKDTa5jHvg>
4. <https://medium.com/mlearning-ai/enter-the-world-of-diffusion-models-4485fb5c5986>

## <Main papers for important diffusion models>

1. Sohl-Dickstein, Jascha, et al. "Deep unsupervised learning using nonequilibrium thermodynamics." International Conference on Machine Learning. PMLR, 2015.
2. Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in Neural Information Processing Systems 33 (2020): 6840-6851.
3. Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In NeurIPS, 2020.
4. Rombach et al., "High-Resolution Image Synthesis with Latent Diffusion Models", CVPR 2022.
5. Radford, Alec, et al. "Learning transferable visual models from natural language supervision." International Conference on Machine Learning. PMLR, 2021.
6. Razavi, Ali, Aaron van den Oord, and Oriol Vinyals. "Generating diverse high-resolution images with VQ-VAE." (2019).