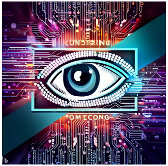
A Quick Tour of Computer Vision

SAST Summer Training 2023

Kevin Zhang

THU CST

July 30th, 2023



1. Introduction
2. Basic Task: Image Classification
3. Recent Progress: Generation Models
4. Reference

Today's Goal

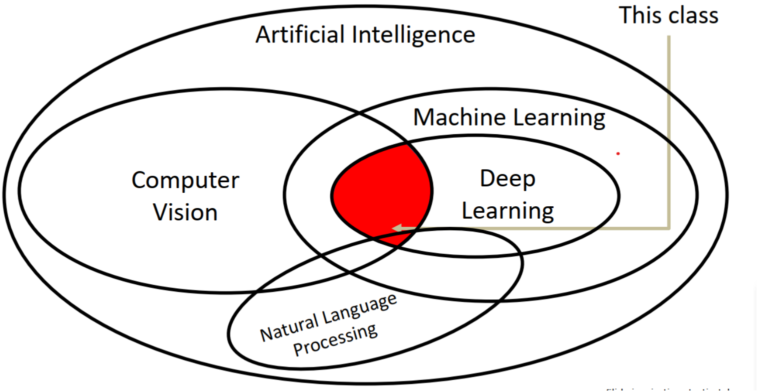
- Explore the applications of previously learned neural networks (e.g. CNN, Transformers) in basic computer vision tasks.
- Understand the fundamentals of cutting-edge generative models: VAE, GANs, and Diffusion Models.
- Explore the widely used open-source diffusion model codebase: *StableDiffusion*.
- Develop your interest and spark your inspiration :)

What's Computer Vision?

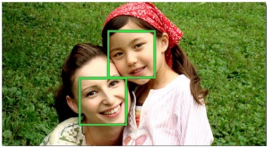
Wiki: Computer vision tasks include methods for **acquiring, processing, analyzing and understanding** digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information.

Note: Computer Vision $\not\subset$ Deep Learning

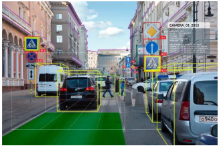
What's Computer Vision?



What's Computer Vision?

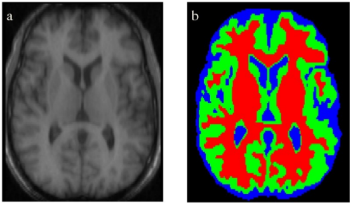


Face detection / recognition

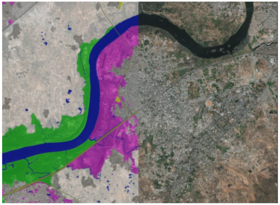


Self-driving cars

Human care



Medical image analysis



Remote sensing / earth observing

What's Computer Vision?

A WIDE range of computer vision tasks:

- Different modalities
  - Image
  - Video
  - 3D Object/Scene
  - Multi-modal (Vision, Language, Speech...)
- Different targets
  - Recognition
  - Segmentation
  - Detection
  - Stylization
  - Captioning
  - Generation
  - ...

Today's Focus

A WIDE range of computer vision tasks:

- Different modalities
  - Image → *Basic*
  - Video
  - 3D Object/Scene
  - Multi-modal (Vision, Language, Speech...) → *Exciting*
- Different targets
  - Recognition → *Basic*
  - Segmentation
  - Detection
  - Stylization
  - Captioning
  - Generation → *Exciting*
  - ...

Why Image Classification?

- Core computer vision task: make computer perceive the world
- Building blocks for other tasks: detection, captioning, and even AlphaGo...
- The first challenge that deep learning made great success over classical methods
- A good chance to review and get a deeper understanding of previously learned neural network architectures, such as CNN and Transformers

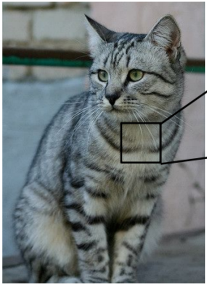
What's Image Classification?

(assume given a set of labels)
{dog, cat, truck, plane, ...}

cat
dog
bird
deer
truck

Challenge for Image Classification

The Problem: Semantic Gap




```
[[105 112 108 111 104  99 106  99  96 103 112 119 104  97  93  87]
 [ 91  98 102 106 104  79  98 103  99 105 123 136 110 105  94  85]
 [ 76  85  90 105 128 105  87  96  95  99 115 112 106 103  99  85]
 [ 99  81  81  93 120 131 127 100  95  98 102  99  96  93 101  94]
 [106  91  61  64  69  91  88  85 101 107 109  98  75  84  96  95]
 [114 108  85  55  55  69  64  54  64  87 112 129  98  74  84  91]
 [133 137 147 103  65  81  80  65  52  54  74  84 102  93  85  82]
 [128 137 144 140 109  95  86  70  62  65  63  63  60  73  86 101]
 [125 133 148 137 119 121 117  94  65  79  80  65  54  64  72  98]
 [127 125 131 147 133 127 126 131 111  96  89  75  61  64  72  84]
 [115 114 109 123 150 148 131 118 113 109 100  92  74  65  72  78]
 [ 89  93  90  97 108 147 131 118 113 114 113 109 106  95  77  80]
 [ 63  77  86  81  77  79 102 123 117 115 117 125 125 130 115  87]
 [ 62  65  82  89  78  71  80 101 124 126 119 101 107 114 131 119]
 [ 63  65  75  88  89  71  62  81 120 138 135 105  81  98 110 118]
 [ 87  65  71  87 106  95  69  45  76 130 126 107  92  94 105 112]
 [118  97  82  86 117 123 116  66  41  51  95  93  89  95 102 107]
 [164 146 112  80  82 120 124 104  76  48  45  66  88 101 102 109]
 [157 170 157 120  93  86 114 132 112  97  69  55  70  82  99  94]
 [130 128 134 161 139 100 109 118 121 134 114  87  65  53  69  86]
 [128 112  96 117 150 144 120 115 104 107 102  93  87  81  72  79]
 [123 107  96  86  83 112 153 149 122 109 104  75  80 107 112  99]
 [122 121 102  80  82  86  94 117 145 148 153 102  58  78  92 107]
 [122 164 148 103  71  56  78  83  93 103 119 139 102  61  69  84]]
```

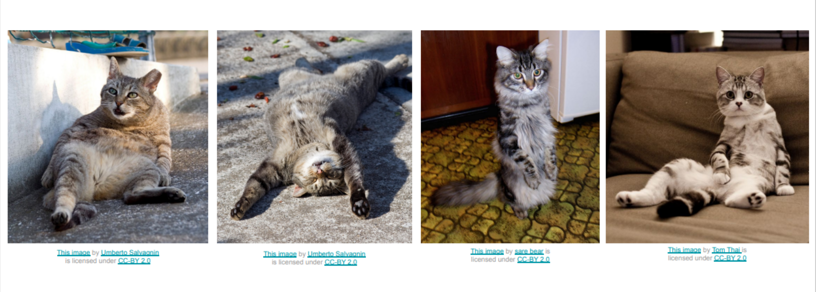
What the computer sees

An image is a tensor of integers between [0, 255]:

e.g. 800 x 600 x 3
(3 channels RGB)

Challenge for Image Classification

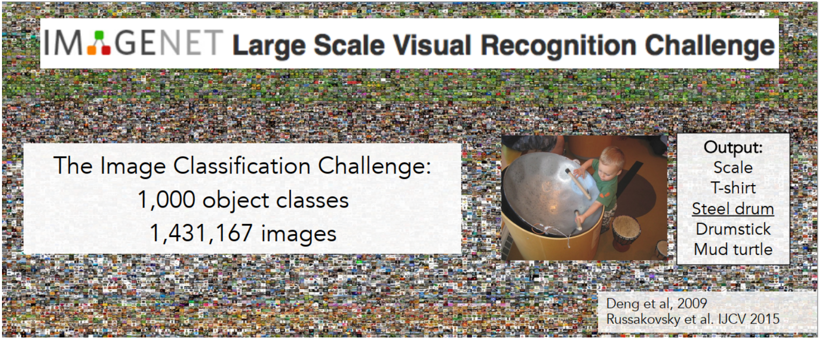
So many different cats!



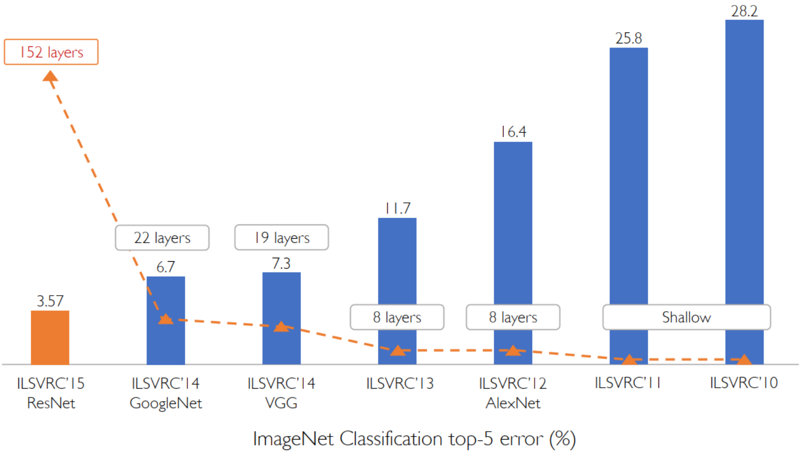
Challenge for Image Classification

So many different cats (or cat tails)!

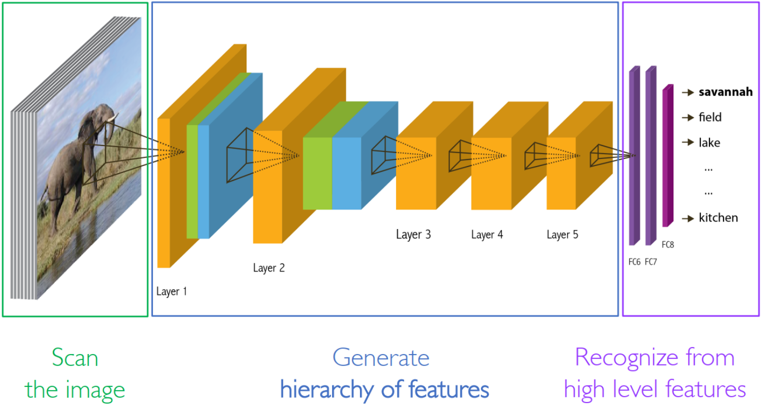
Opportunity: Big data



Imagenet ILSVRC

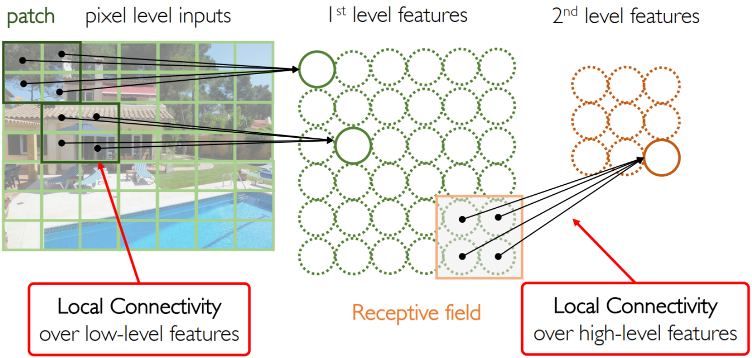


Recap: CNN architecture



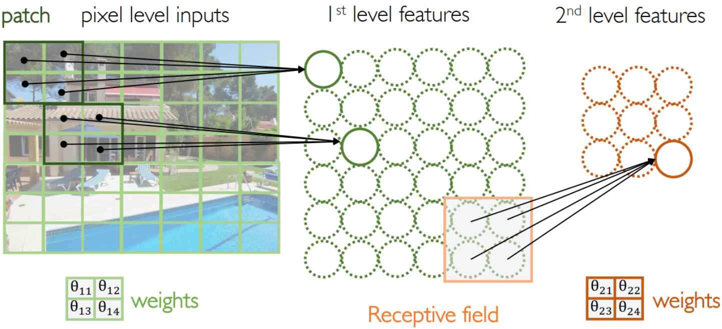
Extract features: convolution layers + pooling layers + normalization layers

Recap: Local Connectivity in CNN



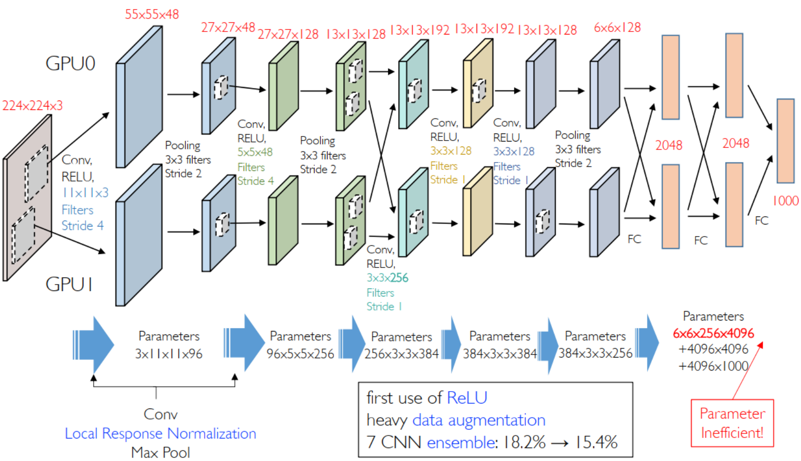
Key assumption 1 (Locality): Assume local information is enough for feature extraction and recognition.

Recap: Parameter Sharing in CNN



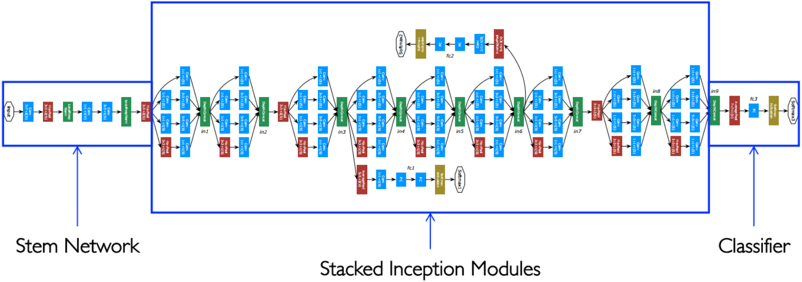
Key assumption 2 (Shift invariance): If a feature is useful at one position, then it should also be useful at other positions.

AlexNet (2012)



GoogLeNet (Inception, 2014): Going deeper

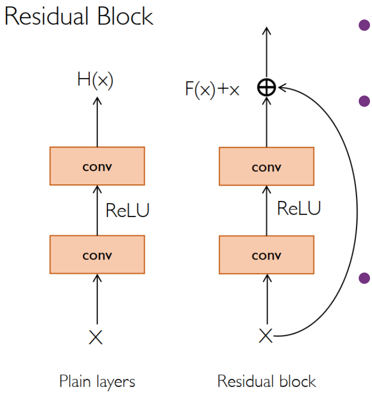
Deeper networks, with computational efficiency



- 22 layers
- Efficient Inception modules
- The first one without FC layers

- Only 5 million parameters! 12x less than AlexNet
- ILSVRC'14 classification winner (6.7% top 5 error)

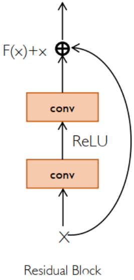
ResNet (CVPR best paper, 2015): Going really deep

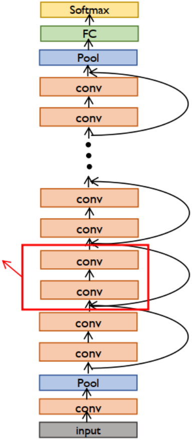


- Key observation: Deeper models are hard to optimize.
- Idea:
  - Copying the learned layer from the shallow model
  - Setting additional layers to identical mapping
- Solution: Instead of learning $H(x)$ directly, learn the residual $F(x) = H(x) - x$.

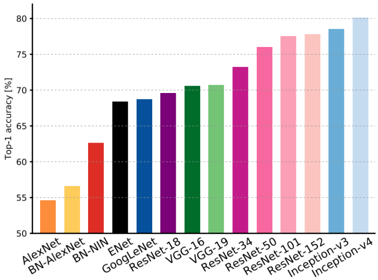
ResNet (CVPR best paper, 2015): Going really deep

- ResNet architecture:
  - Stack residual blocks
  - Every residual block has two 3x3 conv layers
  - Periodically, double #filters and downsample spatially using stride 2 (/2 in each dimension)
  - Additional conv layer at the beginning
  - Global average pooling at the end (FC layer only to output classes)

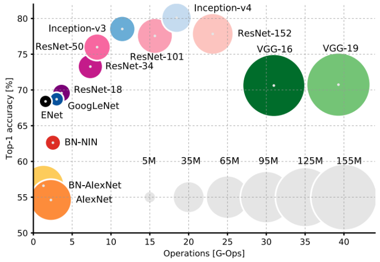




Well-known CNN models that emerged between 2012 and 2017



Top1 vs. Architectures

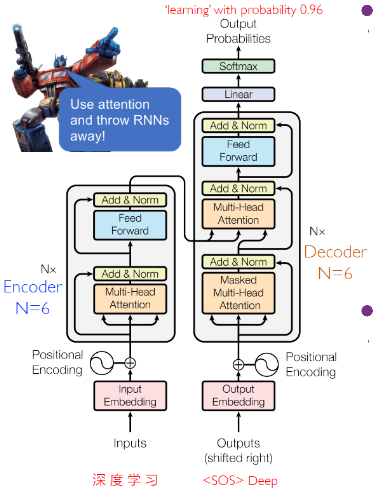


Top1 vs. Operations, Size $\propto$ Parameters

搭积木？知乎：像 ResNet、SENet 这些网络是怎么想出来的？

1. Introduction
2. Basic Task: Image Classification
   - Introduction
   - CNN for Image Classification
   - Vision Transformer (ViT): Towards Larger Model
3. Recent Progress: Generation Models
4. Reference

Recap: Transformer

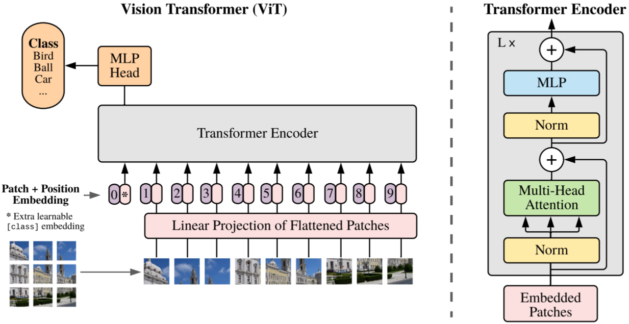


- Main Components:
  - Scaled Dot-product Attention
  - (Masked) Multi-head Attention
  - Position-wise FFN
  - Residual Connections
  - Layer Normalization
  - Positional Encoding
- Architecture:
  - Encoder $\rightarrow$
    A new way to extract features!
  - Decoder with Masking
  - Encoder-Decoder Attention

ViT: An image is worth 16*16 words

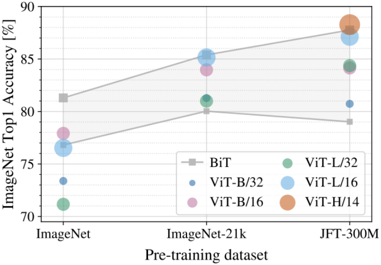
Simple idea: Split the image into fixed-size patches and treat the image as a sequence of patches! (The same Transformer Encoder architecture as before)

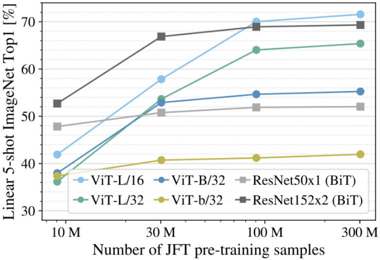
$$\mathbf{x}^{H\times W\times C} \to \mathbf{x}_p^{N\times(P^2\cdot C)} \to \mathbf{z}^{N\times D}$$



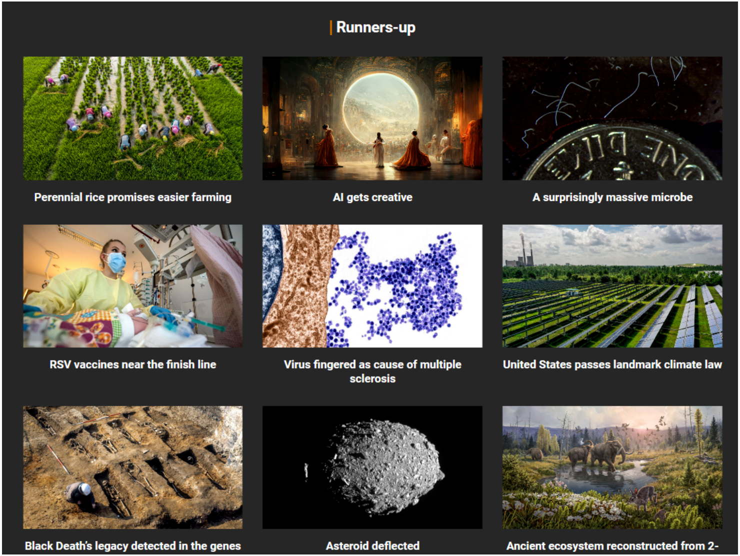
ViT: An image is worth 16*16 words

- Less inductive bias → One architecture for **multi-modal data** and **multiple downstream tasks**!
- Easy to **scale up**! (Demanding for extremely large scale dataset, model regularization and data augmentation)





AIGC: A new opportunity



Transformer again: Unify texts and images

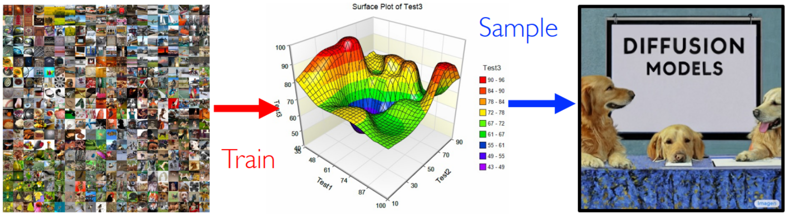
Challenges for applying GPT-like large models to image generation:

- Huge computation consumption if generating in pixel space (A single image might have tens of thousands of pixels)
- Two-dimensional information is probably ignored in sequence generation

Initial solution: Reduce dimensions and generate in the latent space

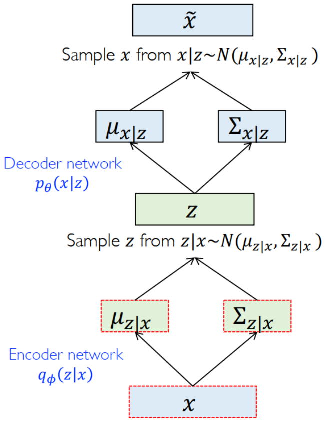
Generative Modeling

Learn the probability distribution $p_\theta(x)$ that generates the data.



Data: $\{(x_i)\}_{i=1}^{n}$ → **Generative Modeling** $f(x)$ → Distribution: $P_\theta(x)$

Variational Auto-Encoder (VAE)



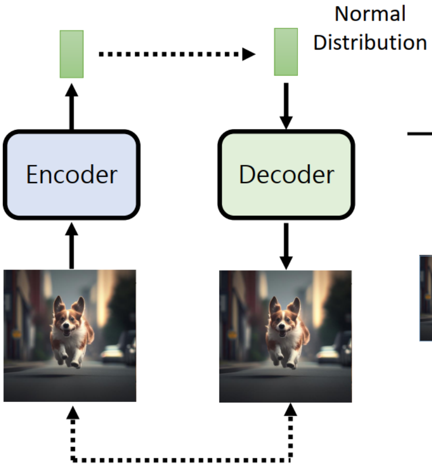
Training:

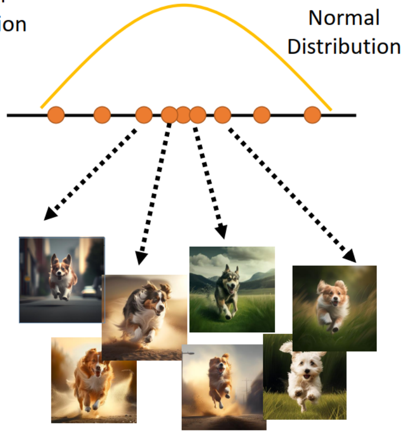
- **Encoder** output should match the prior distribution, e.g. $\mathcal{N}(0, 1)$.
- **Decoder** output should reconstruct the input distribution.

Inference:

- Throw the encoder away, sample from the prior distribution and use the decoder to get output image.

Variational Auto-Encoder (VAE)



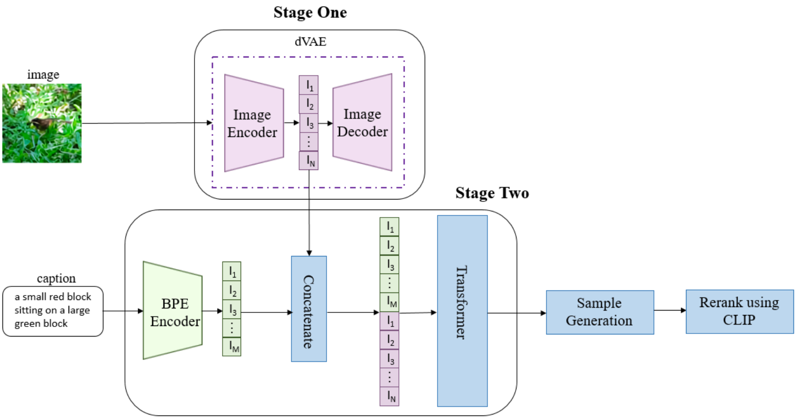


You want more math? (For those interested, not required)

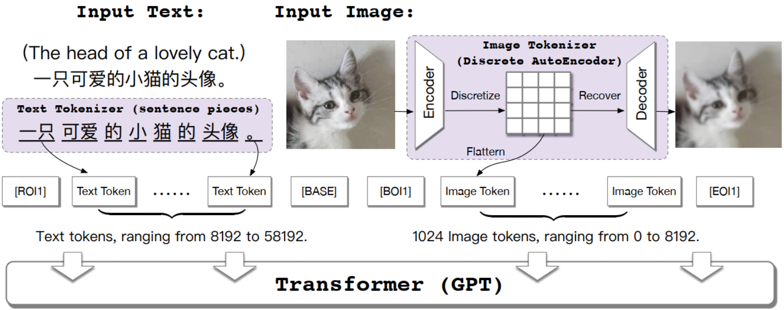
Goal: Maximize $p_\theta(x) \rightarrow$ Objective: Evidence lower Bound, ELBO (θ: Decoder parameters, ϕ: Encoder parameters)

$$
\begin{aligned}
\log p_\theta(x_i) &= \mathbb{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i)] \\
&= \mathbb{E}_z[\log \frac{p_\theta(x_i|z)p_\theta(z)}{p_\theta(z|x_i)}] \\
&= \mathbb{E}_z[\log p_\theta(x_i|z)\frac{p_\theta(z)}{q_\phi(z|x_i)}\frac{q_\phi(z|x_i)}{p_\theta(z|x_i)}] \\
&= \mathbb{E}_z[\log p_\theta(x_i|z)] - \mathbb{E}_z[\log \frac{q_\phi(z|x_i)}{p_\theta(z)}] + \mathbb{E}_z[\log \frac{q_\phi(z|x_i)}{p_\theta(z|x_i)}] \\
&= \mathbb{E}_z[\log p_\theta(x_i|z)] - KL(q_\phi(z|x_i)||p_\theta(z)) \\
&\quad + KL(q_\phi(z|x_i)||p_\theta(z|x_i)) \\
&\geq \mathbb{E}_z[\log p_\theta(x_i|z)] - KL(q_\phi(z|x_i)||p_\theta(z))
\end{aligned}
$$

DALL-E (OpenAI)



CogView (Tsinghua)



The framework of CogView. [ROI1], [BASE1], etc., are seperator tokens.

Pros and cons of VAE

Pros:

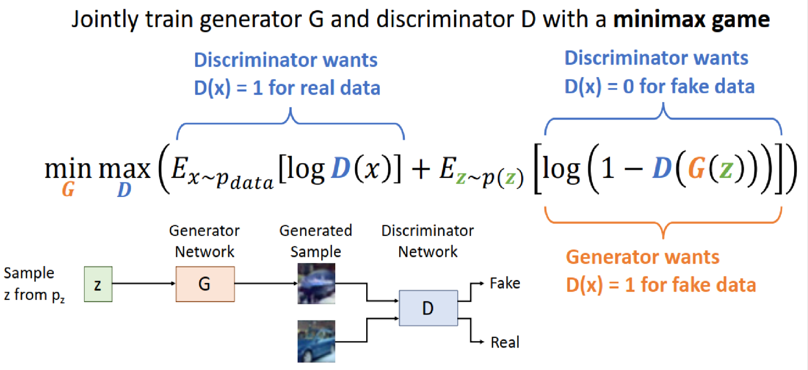
- Principled approach to generative models with solid mathematics basis.
- Allow inference of encoder $q_\phi(z|x_i)$, can be useful low-dimension feature representation for other tasks and models.

Cons:

- Evidence Lower bound: an over-simplified approximation.
- Intractable for complex distributions.
- Blurrier and lower quality than GANs and Diffusion models.

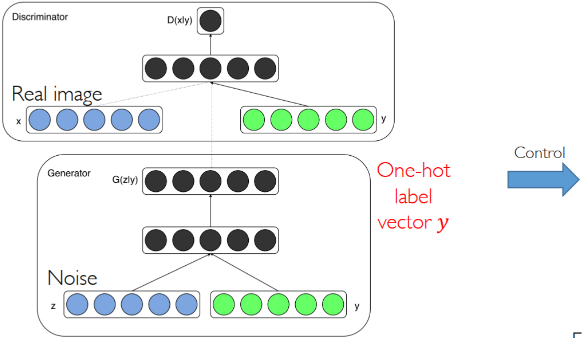
1. Introduction
2. Basic Task: Image Classification
3. Recent Progress: Generation Models
   - Transformer-based Auto-regressive Models
   - Generative Adversarial Networks (GANs)
   - Diffusion Models
4. Reference

GAN: A two-player game



- The center of GAN is an adversarial loss! $\rightarrow$ An idea easy to generalize.

Conditional GAN



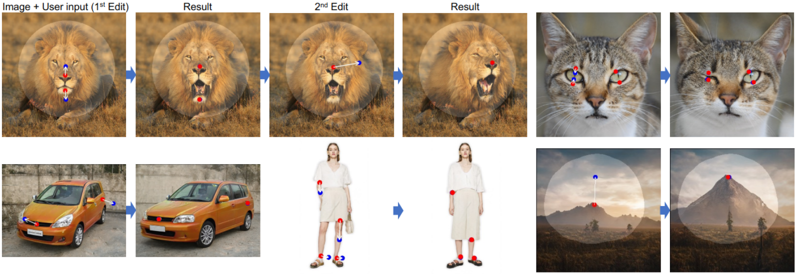
Each row shares same label

- Provide condition information to indicate what to generate.

StyleGAN

- Latent $\mathbf{z} \to$ style features, noise $\to$ content features (diverse details).

DragGAN (SIGGRAPH 2023)



- Utilizing the continuous and discriminative latent space of *StyleGAN* to apply natural and meaningful editing.

Pros and cons of GAN

Pros:

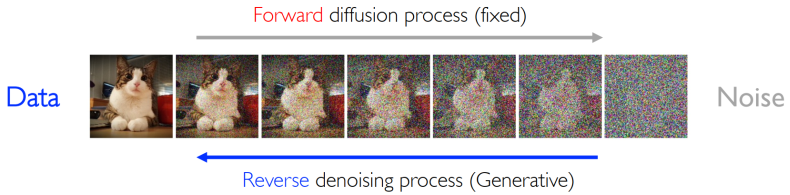
- High-quality generated images (enough to fool the discriminator and also your eyes!)
- High sampling efficiency!
- Relatively continuous latent space (which stores meaningful semantic information).

Cons:

- Hard to train! (A minimax game) $\rightarrow$ Relatively difficult to scale up.
- Mode collapse (Relatively low diversity) $\rightarrow$ A lot of follow-up works to mitigate the problem, e.g. *WGAN*.

1. Introduction
2. Basic Task: Image Classification
3. Recent Progress: Generation Models
   - Transformer-based Auto-regressive Models
   - Generative Adversarial Networks (GANs)
   - Diffusion Models
4. Reference

Denoising Diffusion Probabilistic Model (DDPM)

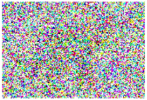


Denoising diffusion models consist of two processes:

- **Forward diffusion process**: Adds noise to input gradually.
- **Reverse denoising process**: learns to generate by denoising.

Training

x_0: clean image



ε: noise

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ ← sample clean image
3: $t \sim \mathrm{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ← sample a noise
5: Take gradient descent step on

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\underbrace{\boxed{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}}_{\text{Noisy image}}, t\right) \right\|^2$$

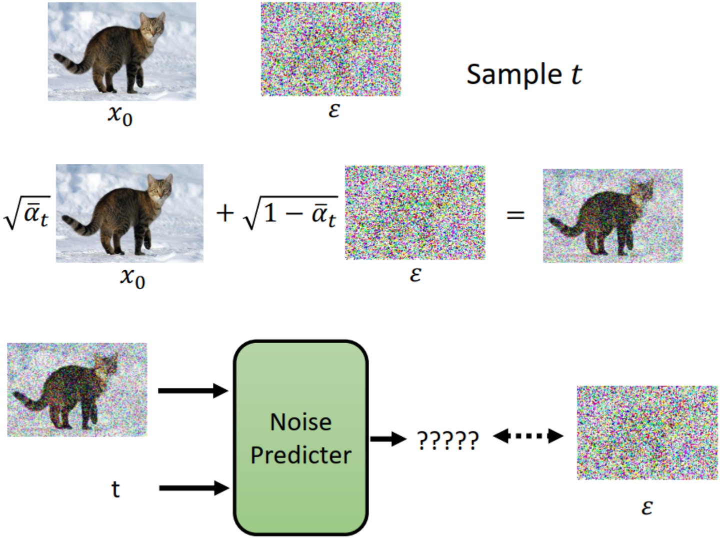
6: **until** converged

$\bar{\alpha}_1, \bar{\alpha}_2, \dots \bar{\alpha}_T \xrightarrow{}$ smaller

Target Noise

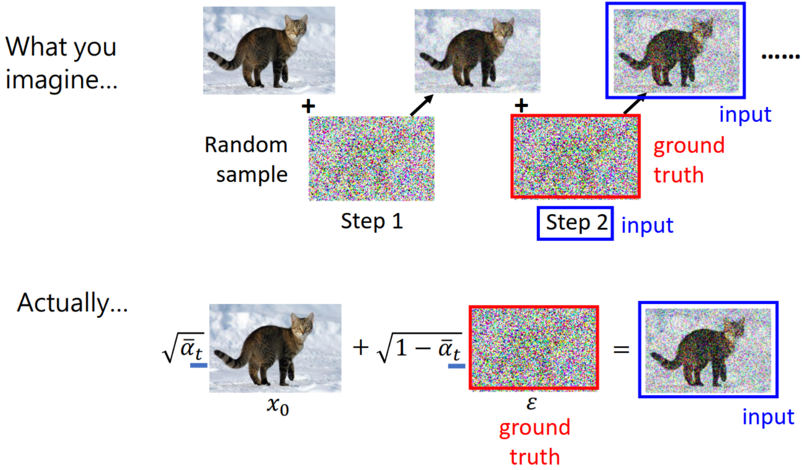
Noise predictor

Training

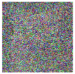


x_0
ε
Sample t
$\sqrt{\bar{\alpha}_t}$
x_0
$+\sqrt{1-\bar{\alpha}_t}$
ε
$=$
Noise Predicter
?????
t
ε

Training


What you imagine...
Random sample
+
Step 1
+
ground truth
input
Step 2
input
......
Actually...
$\sqrt{\bar{\alpha}_t}$
x_0
$+ \sqrt{1-\bar{\alpha}_t}$
ε
ground truth
=
input

Inference



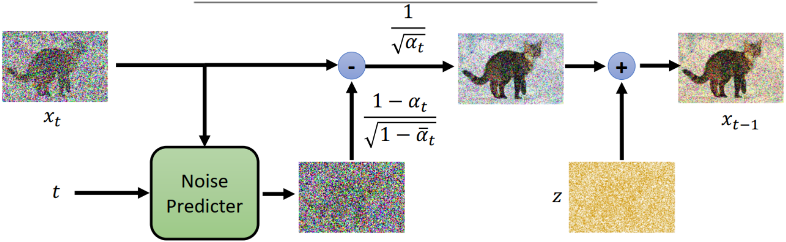
x_T

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

$\bar{\alpha}_1, \bar{\alpha}_2, \ldots \bar{\alpha}_T$

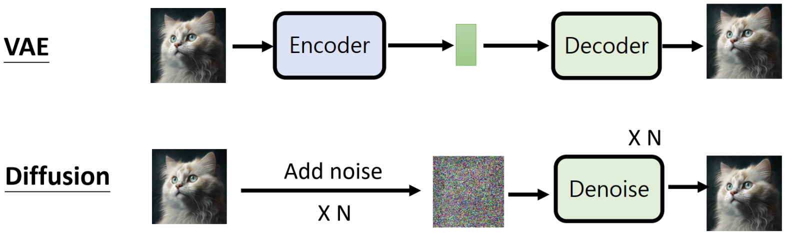
$\alpha_1, \alpha_2, \ldots \alpha_T$



You want more math? (For those interested, not required at all)

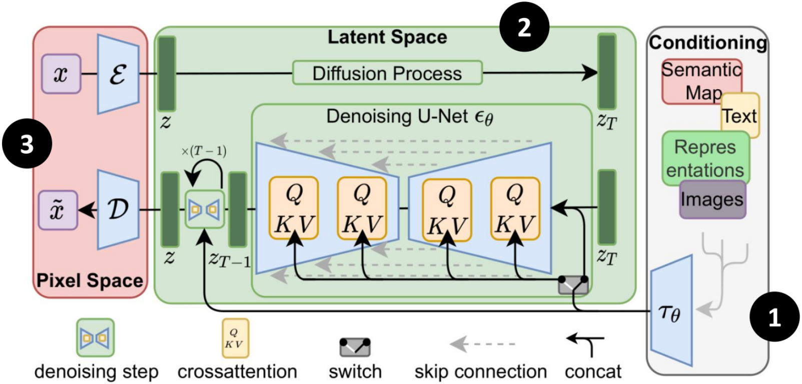
- Lil's blog: What are Diffusion Models?
- 知乎：扩散模型之 DDPM
- My own notes (uploaded in Tsinghua Cloud)

VAE v.s. DDPM



- One step v.s. N steps: Efficiency & Quality tradeoff
- Latent low-dimension representation v.s. Gaussian noise of the same shape

Stable Diffusion: Conditional diffusion in latent space



- Try it on huggingface: `https://huggingface.co/spaces/stabilityai/stable-diffusion`

Something practical: Stable Diffusion open-source codebase

- Github repo: `https://github.com/CompVis/stable-diffusion`
- Built on PyTorch Lightning
- Widely used by mountains of downstream tasks

Usage: Inference

1. Create conda environment:

```
conda env create -f environment.yaml
conda activate ldm
```

2. Download checkpoints from huggingface and put or link it in /*models* folder:

```
mkdir -p models/ldm/stable-diffusion-v1/
ln -s <path/to/model.ckpt>
    models/ldm/stable-diffusion-v1/model.ckpt
```

3. Edit config file in the /*config* folder
4. Sample images:

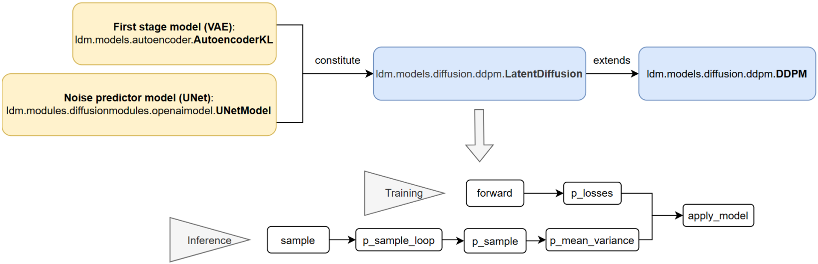
```
python scripts/txt2img.py --prompt "..."
```

Usage: Training

```
python main.py -t -b <config-files> -l <log-file>
```

(The preceding procedure is the same as inference)

Code structrue



Reference and further reading

- All the reference papers are embedded as a hyperlink in the corresponding slide title.
- Some of the pictures in this slide are credited to the following wonderful courses:
    - Stanford University CS231n: Deep Learning for Computer Vision
    - National Taiwan University: Machine Learning by Hung-Yi Lee
    - Tsinghua University: Deep Learning by Mingsheng Long

Thanks!