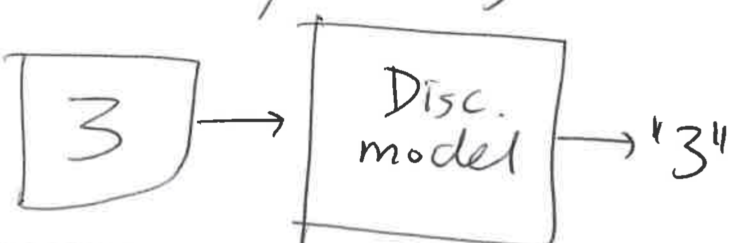
ECE/CS 559 - Neural Networks.

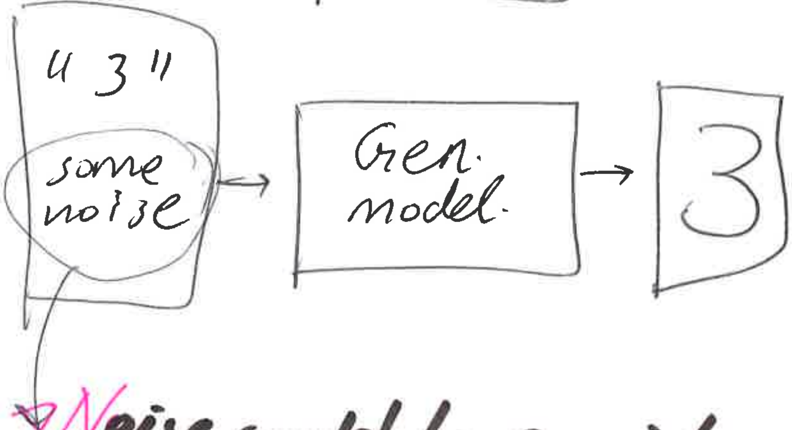
Generative Models.

* Two main approaches in machine learning: generative vs. discriminative approach.
* Observation: X, Target: Y
(e.g. an input image) (e.g. a class label).
* A discriminative model can provide $P(Y|X=x)$
(e.g given the input, what are the likelihoods of different class labels).
* Whereas in ~~Di~~ generative model, we are interested in finding $P(X|Y=y)$
Given label, generate some samples ~~___~~ belonging to that label (kind of an inverse problem):

Discriminative model:



Generative model

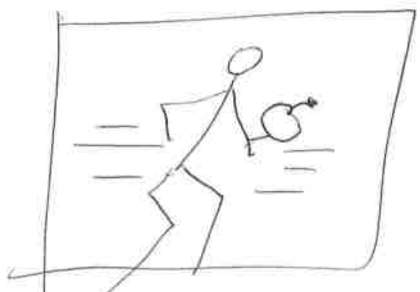


Noise could be considered as a "seed". Different noises would generate different images of 3s.

Applications of generative models

* Low ~~to~~ high resolution image synthesis.
* Text to image translation
  "A guy running with an apple on his hand".

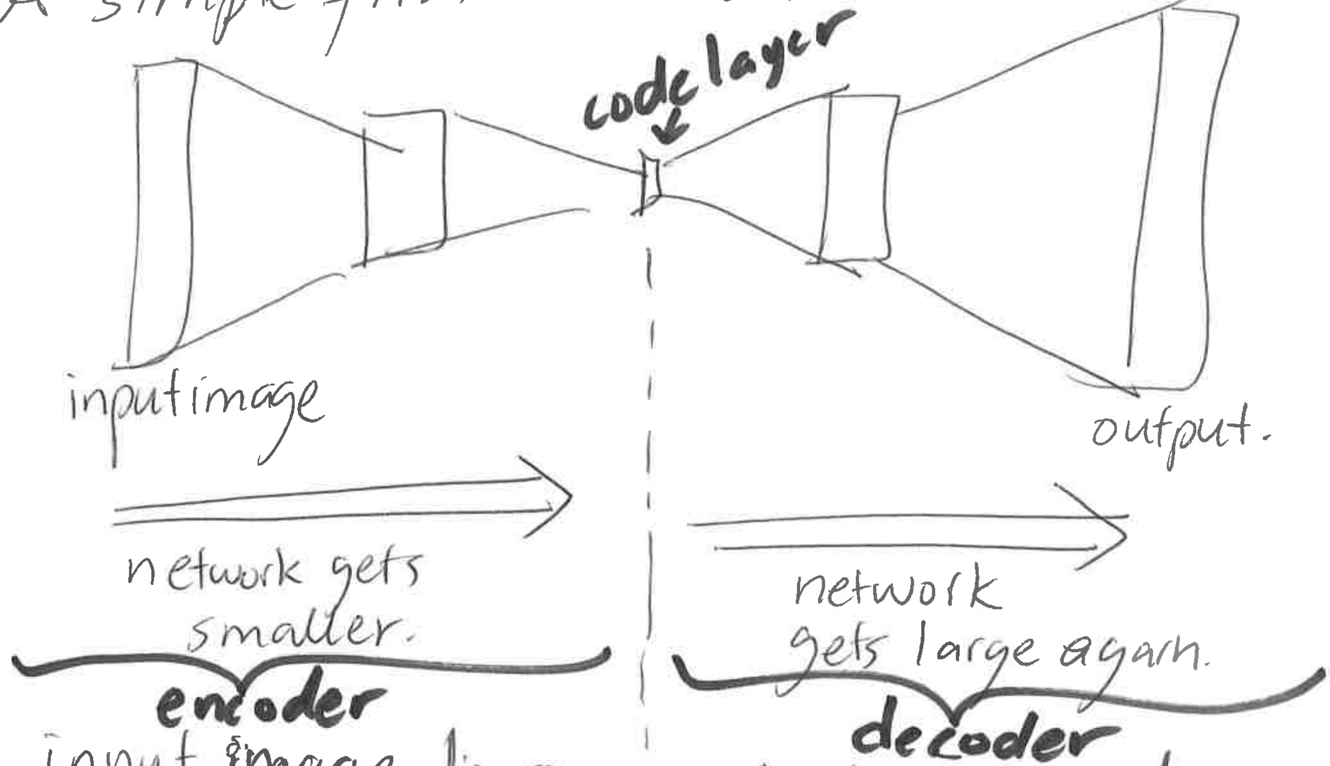
$$\Downarrow$$



* Speech synthesis.
* Error correction

⋮

A simple first idea: Autoencoders.



input image dims = output image dims.

IDEA: Desired output for a given input X

=

Input X.

The network learns to compress / encode X to a very-low-dimensional representation (through the encoder layers) and then reconsruct the original X through the decoder layers.

After training, one can use the decoder part of the network as a "generator".

Generative Adversarial Networks.

introduced by Goodfellow et al 2014.

* **Two main ideas:**

✓ A discriminator network D.

✓ A generator network G.

Ideally $D(x) = \begin{cases} 1, & \text{if } x \text{ is a real image} \\ 0, & \text{if } x \text{ is a fake image provided by e.g. the generator network.} \end{cases}$

In general, $D(x)$ is the likelihood of an image being real.

G takes noise z as an input and generates $G(z)$, a generated image.

* Define the objective

$$\min_{G} \max_{D} \mathbb{E}\left[\log D(x) + \log(1 - D(G(z)))\right]$$

* <u>Theorem</u>: The solution satisfies $f_{G(z)}(x) = f_X(x)$.

Hence, the probability density output of the generator matches the input data distribution.

Proof (Sketch): ① For a fixed G, find the optimal discriminator.

② Optimize over G.

How to train GANs?

1. Generate data samples $x_1, \dots, x_m$
2. Generate noise samples $z_1, \dots, z_m$.
3. Update the discriminator by <u>ascending</u> its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left(\log D(x_i) + \log(1 - D(G(z_i))) \right)$$

θ_d → Discriminator parameters.

4. Generate new noise samples $z_1, \dots, z_m$
5. Update the generator by <u>descending</u> its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D(G(z_i))\right).$$

θ_g → generator parameters

6. Goto (1) until convergence.

At convergence, one arrives at a solution where $f_{G(Z)}(x) = f_X(x)$ and $D(x) = \frac{1}{2}$ $\forall x$. (all images are equally likely to be real or fake as the generator is perfect.)

6 Taken from Goodfellow et al.'s paper.

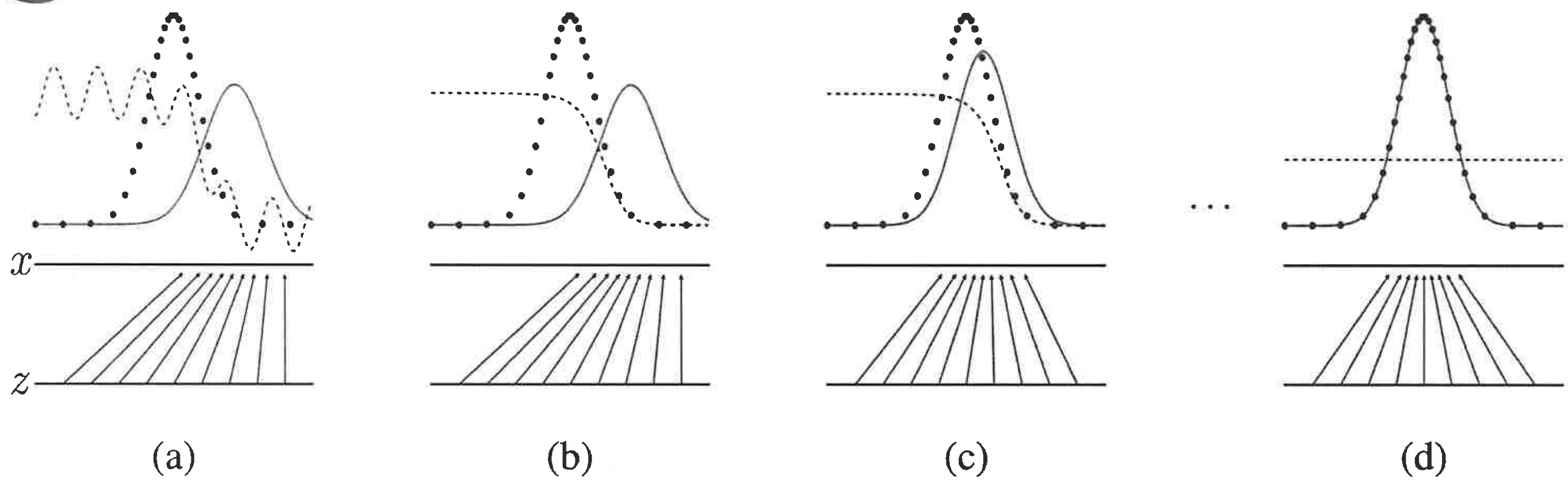


Figure 1: Generative adversarial nets are trained by simultaneously updating the discriminative distribution (D, blue, dashed line) so that it discriminates between samples from the data generating distribution (black, dotted line) $p_{\boldsymbol{x}}$ from those of the generative distribution p_g (G) (green, solid line). The lower horizontal line is the domain from which $\boldsymbol{z}$ is sampled, in this case uniformly. The horizontal line above is part of the domain of $\boldsymbol{x}$. The upward arrows show how the mapping $\boldsymbol{x} = G(\boldsymbol{z})$ imposes the non-uniform distribution p_g on transformed samples. G contracts in regions of high density and expands in regions of low density of p_g. (a) Consider an adversarial pair near convergence: p_g is similar to p_{data} and D is a partially accurate classifier. (b) In the inner loop of the algorithm D is trained to discriminate samples from data, converging to $D^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x})+p_g(\boldsymbol{x})}$. (c) After an update to G, gradient of D has guided $G(\boldsymbol{z})$ to flow to regions that are more likely to be classified as data. (d) After several steps of training, if G and D have enough capacity, they will reach a point at which both cannot improve because $p_g = p_{\text{data}}$. The discriminator is unable to differentiate between the two distributions, i.e. $D(\boldsymbol{x}) = \frac{1}{2}$.

Conditional GANs:

GANs cannot natively generate images for a given label. E.g. "Generate an image of a 3". For this purpose, we can use a conditional GAN. All that has to be done is to feed the class label (e.g. as a one-hot-encoded vector) to the generator as well as the discriminator during training (and, of course, during generation). The class label thus becomes an extra input to both D and G.