· HW 2 Grades out

·

# Generative Models

# Distribution learning

· learn a distribution $P$ over $X$ ; evaluate $P(x)$, $x \in X$

· Sample from distribution

Training Data(CelebA)

Model Samples (Karras et.al., 2018)

4 years of progression on Faces


Brundage et al., 2017

Image credits to Andrej Risteski

# Distribution learning

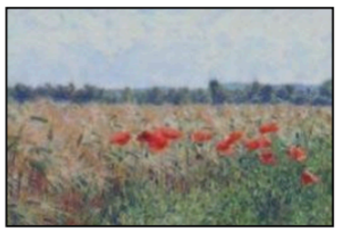
*BigGAN, Brock et al '18*

# Distribution learning

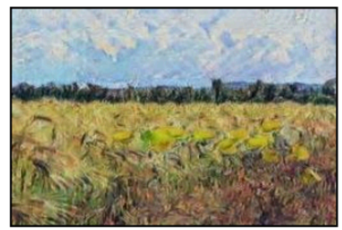
Conditional generative model P(zebra images| horse images)

Style Transfer

Input Image



Monet



Van Gogh

Image credits to Andrej Risteski

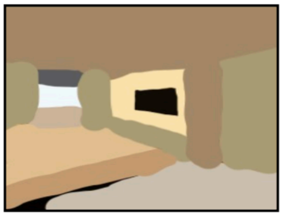
# Distribution learning

Source actor


Real-time reenactment

Target actor

# Generative model



Generate

Generative model
of realistic images

**Stroke paintings to realistic images**
[Meng, He, Song, et al., ICLR 2022]

"Ace of Pentacles"

Generate

Generative model
of paintings

**Language-guided artwork creation**
[https://chainbreakers.kath.io](https://chainbreakers.kath.io) @RiversHaveWings

Slides credit to Yang Song

# Generative model

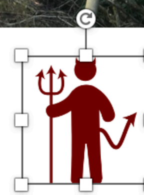
$p(x)$

$p(x)$


High probability

Generative model of traffic signs

Low probability

**Outlier detection**

[Song et al., ICLR 2018]

Slide credit to Yang Song

# Desiderata for generative models

$P_\theta(\cdot)$, $\theta$ is parameter

- **Probability evaluation**: given a sample ($x$), it is computationally efficient to evaluate the probability of this sample.

- **Flexible model family:** it is easy to incorporate any neural network models.

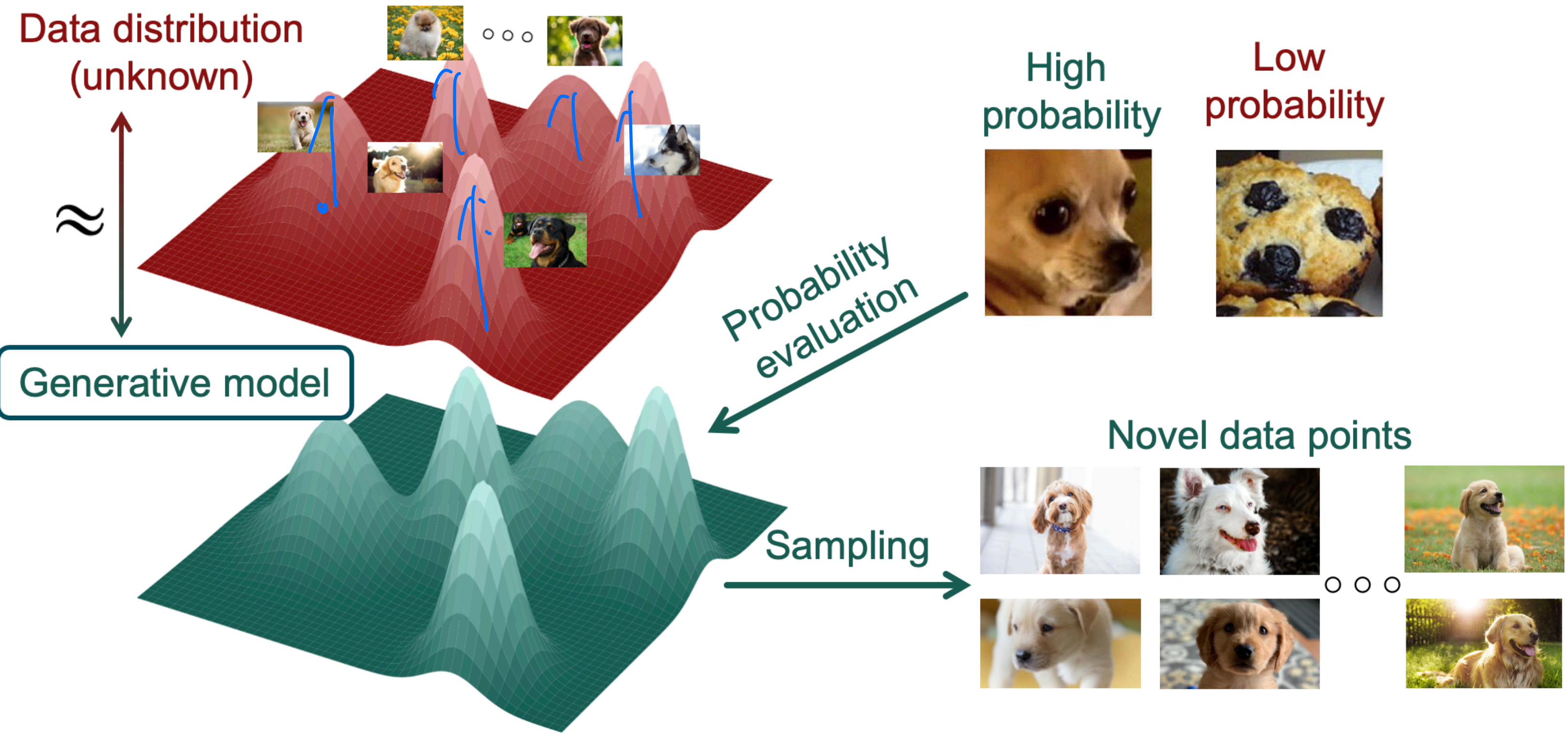
$P_\theta(\cdot)$: CNN, transformer, $P_\theta(x) \geq 0$

supervised. $x \mapsto f(x)$ $\quad \sum_x P_\theta(x) = 1$

- **Easy sampling:** it is computationally efficient to sample a data from the probabilistic model.

$P_\theta(x)$, $\quad x \sim P_\theta(\cdot)$

# Desiderata for generative models




Slide credit to Yang Song

# Taxonomy of generative models

Given x, explicit expression for $p_\theta(x)$

$z \sim N(0, I)$

$x = G_\theta(z)$

$p_\theta(x)$ tractable to evaluate

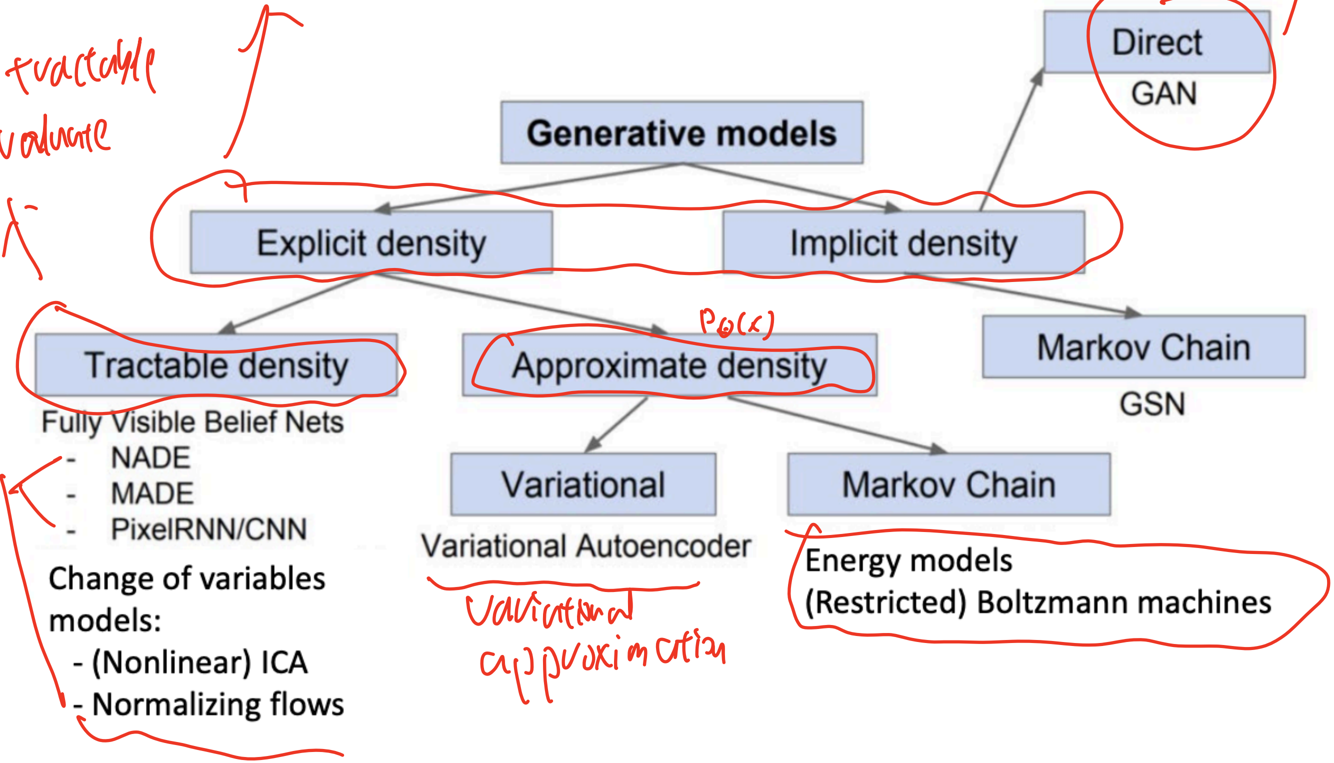
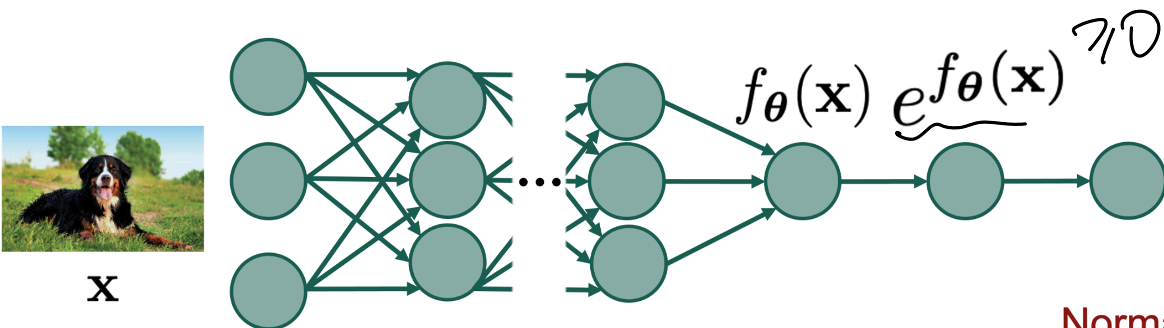


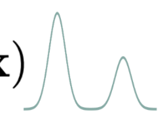

Image credits to Andrej Risteski

# Key challenge for building generative models

distribution $\begin{cases} p_\theta(x) \geq 0 \\ \int_x p_\theta(x)\,dx = 1 \end{cases}$




$$\frac{e^{f_{\boldsymbol{\theta}}(\mathbf{x})}}{\boxed{Z_{\boldsymbol{\theta}}}} = p_{\boldsymbol{\theta}}(\mathbf{x})$$



Normalizing constant

$$\int_x e^{f_\theta(x)}\,dx$$

Slide credit to Yang Song

# Key challenge for building generative models

$p_\theta(x)$

**Approximating the normalizing constant**

- Variational auto-encoders [Kingma & Welling 2014, Rezende et al. 2014]
- Energy-based models [Ackley et al. 1985, LeCun et al. 2006]

✕ Inaccurate probability evaluation

**Using restricted neural network models**

- Autoregressive models [Bengio & Bengio 2000, van den Oord et al. 2016]
- Normalizing flow models [Dinh et al. 2014, Rezende & Mohamed 2015]

✕ Restricted model family

**Generative adversarial networks (GANs)**

- Model the generation process, not the probability distribution [Goodfellow et al. 2014]

✕ Cannot evaluate probabilities

Slide credit to Yang Song

# Training generative models

$\{x_1, \ldots x_n\} \sim p$

$p_\theta(\cdot)$

- **Likelihood-based:** maximize the likelihood of the data under the model (possibly using advanced techniques such as variational method or MCMC):

$$\max_\theta \sum_{i=1}^{n} \log p_\theta(x_i)$$

- Pros:
  - **Easy training**: can just maximize via SGD.
  - **Evaluation**: evaluating the fit of the model can be done by evaluating the likelihood (on test data).

- Cons:
  - **Large models needed**: likelihood objectve is hard, to fit well need very big model.
  - **Likelihood entourages averaging:** produced samples tend to be blurrier, as likelihood encourages "coverage" of training data.

# Training generative models

- **Likelihood-free:** use a **surrogate loss** (e.g., GAN) to train a discriminator to differentiate real and generated samples.
- Pros:
  - **Better objective, smaller models needed:** objective itself is learned - can result in visually better images with smaller models.
- Cons:
  - **Unstable training:** typically min-max (saddle point) problems.
  - **Evaluation:** no way to evaluate the quality of fit.

# Generative Adversarial Nets

# Implicit Generative Model

X $P_\theta(x)$

- **Goal:** a sampler $g(\cdot)$ to generate images
- A simple generator $g(z;\theta)$:
  - $z \sim N(0,I)$
  - $x = g(z;\theta)$ deterministic transformation
- Likelihood-free training:
  - Given a dataset from some distribution $p_{data}$
  - Goal: $g(z;\theta)$ defines a distribution, we want this distribution $\approx p_{data}$
  - Training: minimize $D(g(z;\theta), p_{data})$
    - $D$ is some distance metric (not likelihood)
  - Key idea: ***Learn* a differentiable** $D$: neural net

Metrics

(1) Kullback-Leibler Divergence
KL-Div
(2) Total variation
(3) Wasserstein distance
(4) Jensen-Shannon Divergence
(5) Integral probability metric
(IPM)

# GAN (Goodfellow et al., '14)

binary classification

- Parameterize the discriminator $D(\cdot\,;\phi)$ with parameter $\phi$

- **Goal:** learn $\phi$ such that $D(x;\phi)$ measures how likely $x$ is from $p_{data}$
  - $D(x,\phi) = 1$ if $x \sim p_{data}$
  - $D(x,\phi) = 0$ if $x! \sim p_{data}$
  - a.k.a., a binary classifier
- GAN: use a neural network for $D(\cdot\,;\phi)$

$D(x_i, \phi) \simeq 1$

- **Training:** need both negative and positive samples
  - Positive samples: just the training data $\{x_1, \ldots, x_N\} \sim p_{data}$
  - Negative samples: use our sampler $g(z;\theta)$ (can provide infinite samples).

$D(g(z;\theta), \phi) \simeq 0$

- **Overall objectives:**

Given $\phi$ $\qquad g \simeq p_{data}$

  - Generator: $\theta^* = \max_{\theta} D(g(z;\theta);\phi)$
  - Discriminator uses MLE Training:

$\simeq 0$

$$\phi^* = \max_{\phi} \mathbb{E}_{x\sim p_{data}}[\log D(x;\phi)] + \mathbb{E}_{\hat{x}\sim g(\cdot)}[\log(1 - D(\hat{x};\phi))]$$

# GAN (Goodfellow et al., ‘14)

- Generator $G(z;\theta)$ where $z \sim N(0,I)$
  - Generate realistic data
- Discriminator $D(x;\phi)$
  - Classify whether the data is real (from $p_{data}$) or fake (from $G$)
- Objective function:

$$L(\theta,\phi) = \min_{\theta}\max_{\phi} \mathbb{E}_{x\sim p_{data}}\left[\log D(x;\phi)\right] + \mathbb{E}_{\hat{x}\sim G}\left[\log(1 - D(\hat{x};\phi))\right]$$

$G \approx p_{data}$

$D \approx 1$

- Training procedure:

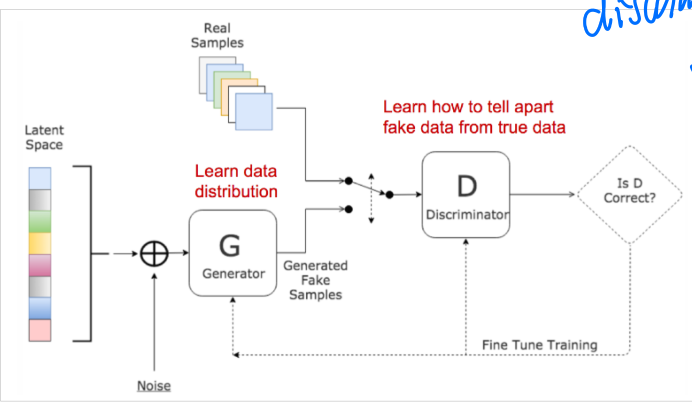
label label

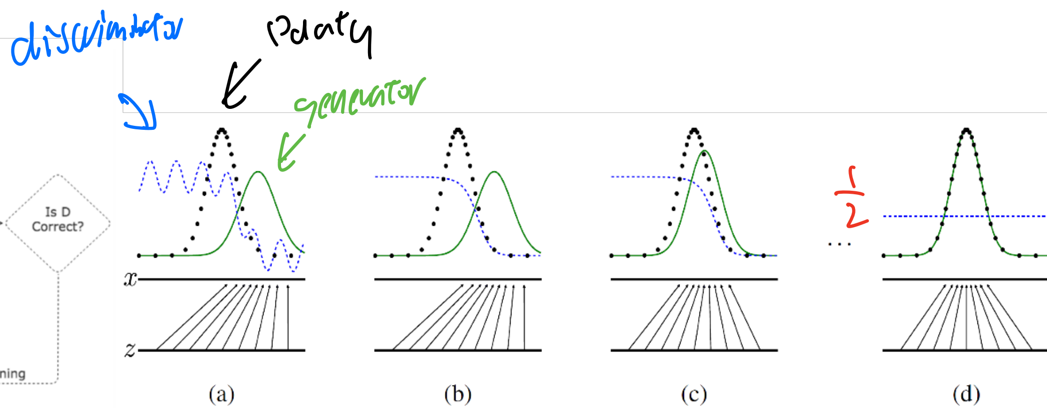
  - Collect dataset $\{(x,1) \mid x \sim p_{data}\} \cup \{(\hat{x},0) \sim g(z;\theta)\}$
  - Train discriminator $D : L(\phi) = \mathbb{E}_{x\sim p_{data}}\left[\log D(x;\phi)\right] + \mathbb{E}_{\hat{x}\sim G}\left[\log(1 - D(\hat{x};\phi))\right]$
  - Train generator $G : L(\theta) = \mathbb{E}_{z\sim N(0,I)}\left[\log D(G(z;\theta),\phi)\right]$
  - Repeat

# GAN (Goodfellow et al., ‘14)

- Objective function:

$$L(\theta, \phi) = \min_{\theta} \max_{\phi} \mathbb{E}_{x \sim p_{data}} \left[ \log D(x; \phi) \right] + \mathbb{E}_{\hat{x} \sim G} \left[ \log(1 - D(\hat{x}; \phi)) \right]$$



# Math Behind GAN

$$L(\theta, \phi) = \min_{\theta} \max_{\phi} \mathbb{E}_{x \sim p_{data}}[\log D(x;\phi)] + \mathbb{E}_{\hat{x} \sim G(\cdot)}[\log(1 - D(\hat{x};\phi))]$$

- Let $D^*$, $G^*$ be the solution to
- Optimal $D^*$, for a given $x$

$$L_x(D) = p_{data}(x) \cdot \log D(x) + P_G(x) \cdot \log(1 - D(x))$$

$$\Rightarrow D^*: \frac{\partial L}{\partial D} = 0 \quad \text{first-order condition}$$

$$\Rightarrow \frac{p_{data}(x)}{D^*(x)} - \frac{p_G(x)}{1 - D^*(x)} = 0, \quad \forall x$$

$$\Downarrow$$

$$\Rightarrow D^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_G(x)}$$

$$\Rightarrow \text{if } p_G = p_{data} \Rightarrow D^*(x) = 0.5$$

# Math Behind GAN

Consider optimal generator $G^*$ given optimal $D^*$

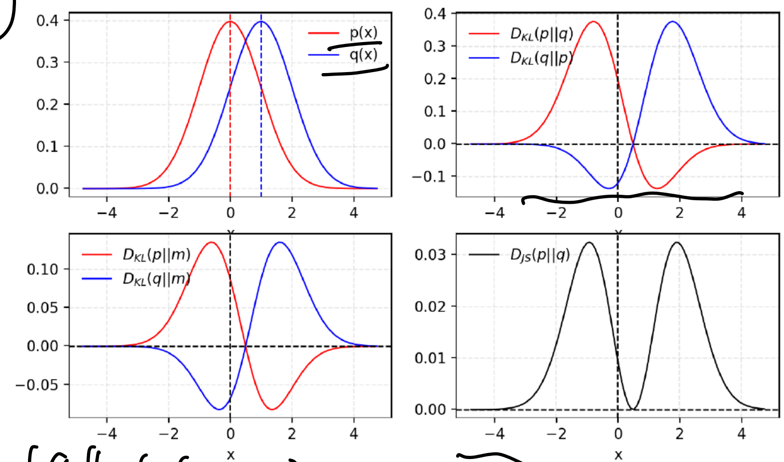
$$L(\theta,\phi) = \mathbb{E}_{x\sim p_{data}}\left[\frac{p_{data}(x)}{p_{data}(x)+p_G(x)}\right] + \mathbb{E}_{\hat{x}\sim G(\cdot)}\left[\log\frac{p_G(\hat{x})}{p_{data}(\hat{x})+p_G(\hat{x})}\right]$$

$$= \mathbb{E}_{x\sim p_{data}}\left[\log\frac{p_{data}(x)}{\frac{p_{data}(x)+p_G(x)}{2}}\right] + \mathbb{E}_{\hat{x}\sim G(\cdot)}\left[\log\frac{p_G(\hat{x})}{\frac{p_{data}(\hat{x})+p_G(\hat{x})}{2}}\right] - \log 4$$

$$= \underbrace{KL\left(p_{data} \,\middle\|\, \tfrac{1}{2}(p_{data}+p_G)\right) + KL\left(p_G \,\middle\|\, \tfrac{1}{2}(p_{data}+p_G)\right)}_{2\,\cdot\,\text{Jensen-Shannon Divergence (JSD)}}$$

# KL-Divergence and JS-Divergence

- $KL(p\|q) = \mathbb{E}_{x\sim p}\left[\log\left(\frac{p(x)}{q(x)}\right)\right]$
  - asymmetric
    $KL(p\|q) \neq KL(q\|q)$




- $JSD(p\|q)$
  $= \frac{1}{2}\left[KL\left(p\|\frac{1}{2}(p+q)\right) + KL\left(q\|\frac{1}{2}(p+q)\right)\right]$
  - symmetric, $JSD \geq 0$, $JSD(p\|q)=0 \Leftrightarrow p=q$
  - $\sqrt{JSD}$ satisfies triangle ineq

# Math Behind GAN

$\Rightarrow$ Given optimal $D^*$

$$L(\theta) = 2 \cdot JSD\left(P_G \,\middle\|\, \frac{1}{2}(P_G + P_{data})\right) - \log 4$$

goal: find $P_G$ minimize $L(\theta)$

$$JSD \geq 0$$

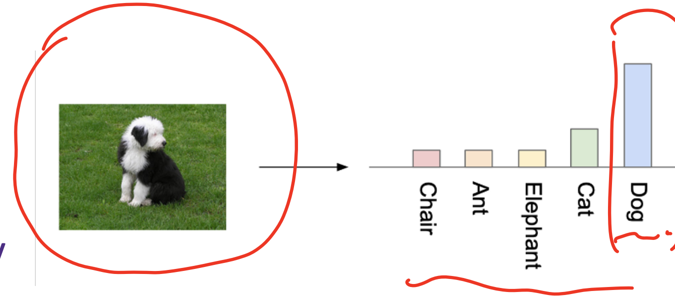
$\Rightarrow$ minimizer $JSD\left(P_G \,\middle\|\, \frac{1}{2}(P_G + P_{data})\right) = 0$

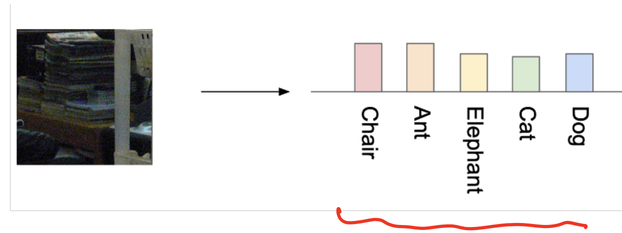
$\Rightarrow$ $P_G = P_{data}$

- $L^* = -\log 4$

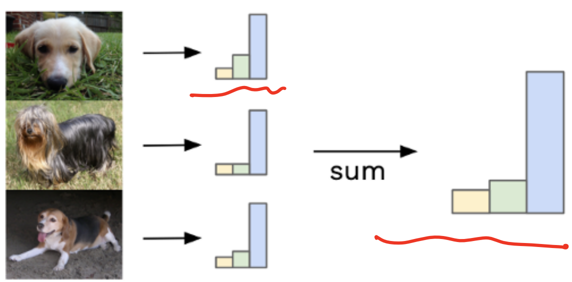
# Evaluation of GAN

- No $p(x)$ in GAN.
- Idea: use a trained classifier $f(y \mid x)$:
- If $x \sim p_{data}$, $f(y \mid x)$ should have low entropy
    - Otherwise, $f(y \mid x)$ close to uniform.
- Samples from $G$ should be diverse: on average marginal
    - $p_f(y) = \mathbb{E}_{x \sim G}[f(y \mid x)]$ close to uniform.

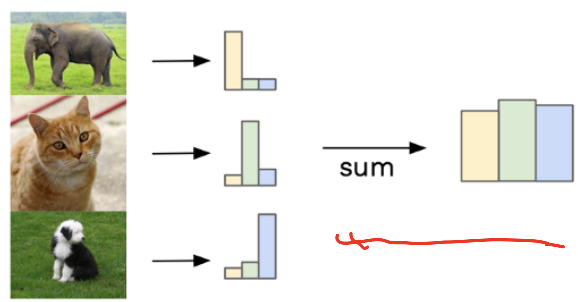



Similar labels sum to give focussed distribution

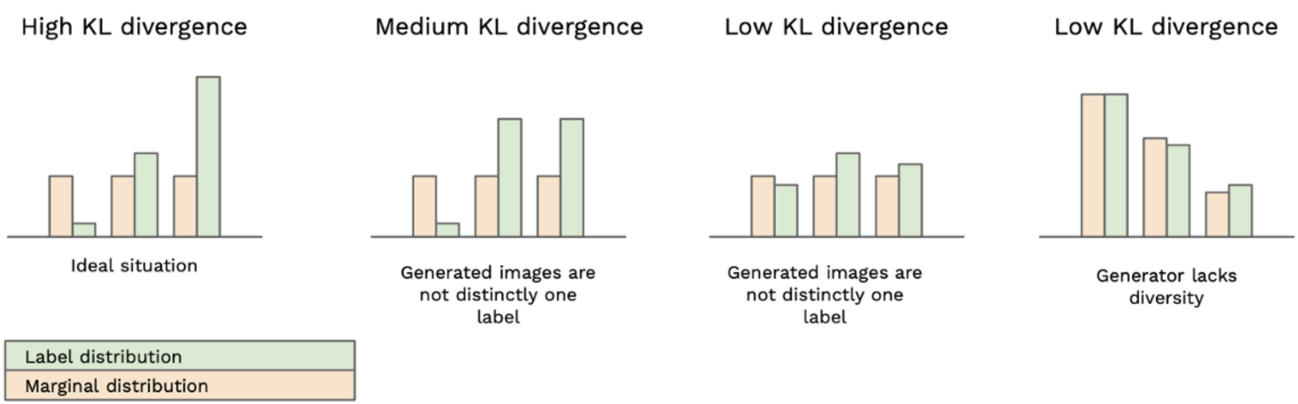



Different labels sum to give uniform distribution



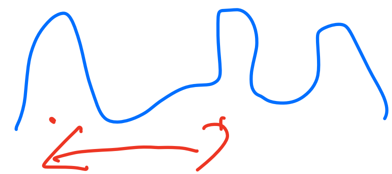
# Evaluation of GAN

- **Inception Score** (IS, Salimans et al. '16)
  - Use Inception V3 trained on ImageNet as $f(y \mid x)$
  - $IS = \exp\left(\mathbb{E}_{x \sim G}\left[KL(\underbrace{f(y \mid x)}_{\text{low-entropy}} \,||\, \underbrace{p_f(y)}_{\text{marginal high entropy}})\right]\right)$
  - Higher the better



# Comments on GAN

- Other evaluation metrics:
  - Fréchet Inception Distance (FID): Wasserstein distance between Gaussians
- Mode collapse:
  - The generator only generate a few type of samples.
  - Or keep oscillating over a few modes.



- Training instability:
  - Discriminator and generator may keep oscillating
  - Example: $-xy$, generator $x$, discriminator$y$. NE: $x = y = 0$ but GD oscillates.
  - No stopping criteria.
  - Use Wsserstein GAN (Arjovsky et al. '17):

$$\min_{G} \max_{f:\text{Lip}(f)\leq 1} \mathbb{E}_{x\sim p_{data}}\left[f(x)\right] - \mathbb{E}_{\hat{x}\sim p_G}[f(\hat{x})]$$

  - And need many other tricks…