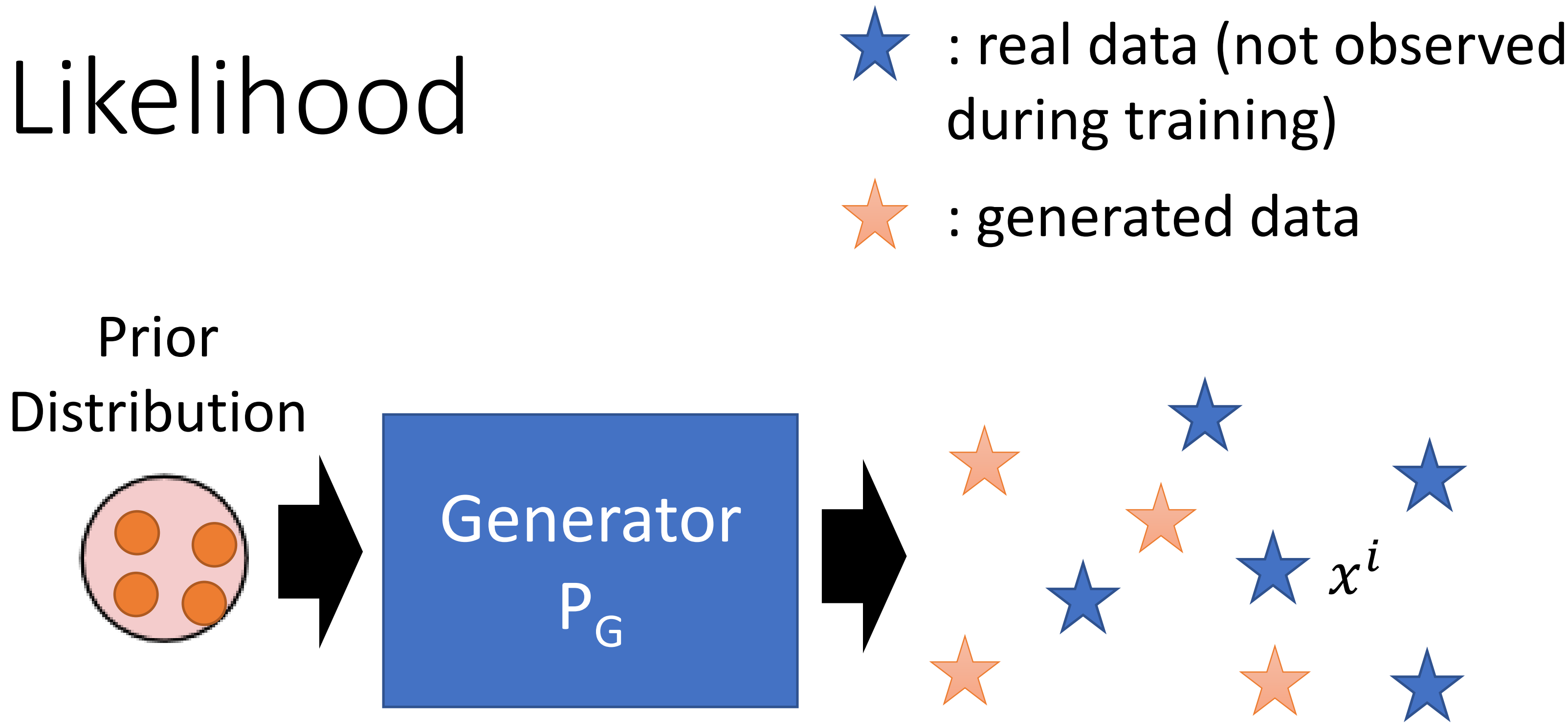
Evaluation

Ref: Lucas Theis, Aäron van den Oord, Matthias Bethge, “A note on the evaluation of generative models”, arXiv preprint, 2015

Likelihood

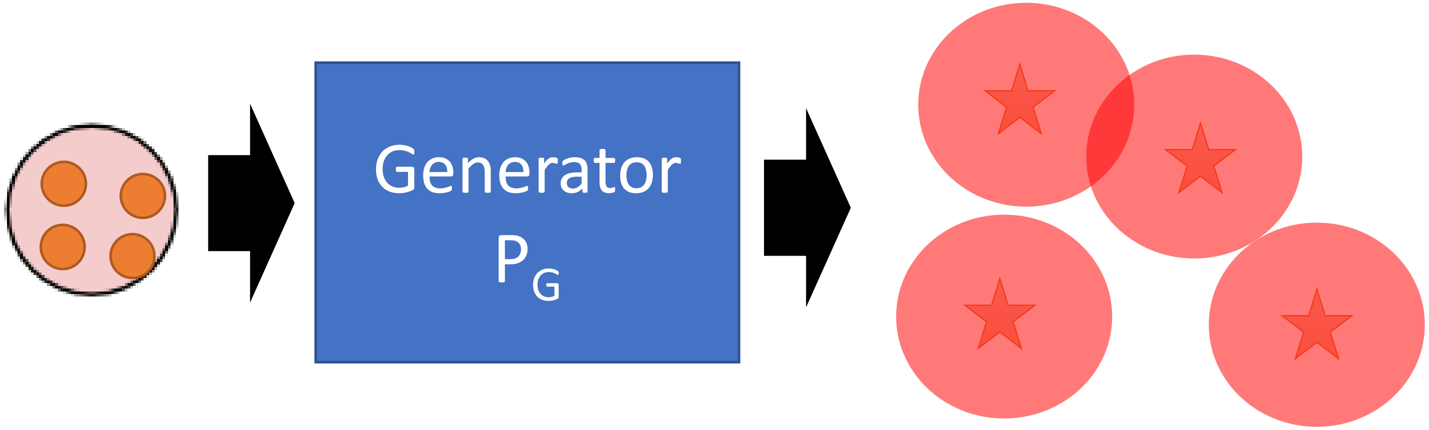


Log Likelihood: $L = \frac{1}{N}\sum_i logP_G(x^i)$

We cannot compute $P_G(x^i)$. We can only sample from P_G.

Likelihood
- Kernel Density Estimation

- Estimate the distribution of $P_G(x)$ from sampling



Each sample is the mean of a Gaussian with the same covariance.

Now we have an approximation of P_G, so we can compute $P_G(x^i)$ for each real data x^i

Then we can compute the likelihood.

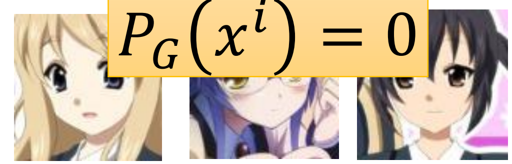
Likelihood v.s. Quality

- Low likelihood, high quality?

Considering a model generating good images (small variance)

Generator P_G

Generated data



$P_G(x^i) = 0$

Data for evaluation x^i

- High likelihood, low quality?

X 0.99

G1

G2

X 0.01

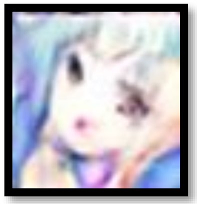
$$L = \frac{1}{N}\sum_i \underbrace{logP_G(x^i)}_{100} = \underbrace{-log100}_{4.6} + \frac{1}{N}\sum_i logP_G(x^i)$$

[Tim Salimans, et al., NIPS, 2016]

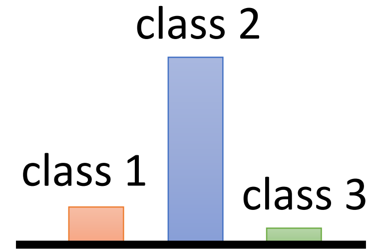
Objective Evaluation

x: image

y: class (output of CNN)

x  → Off-the-shelf Image Classifier → $P(y|x)$

e.g. Inception net, VGG, etc.



Concentrated distribution means higher visual quality

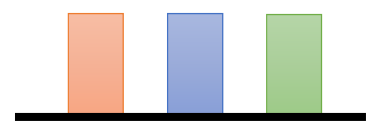
x^1 → CNN → $P(y^1|x^1)$

x^2 → CNN → $P(y^2|x^2)$

x^3 → CNN → $P(y^3|x^3)$

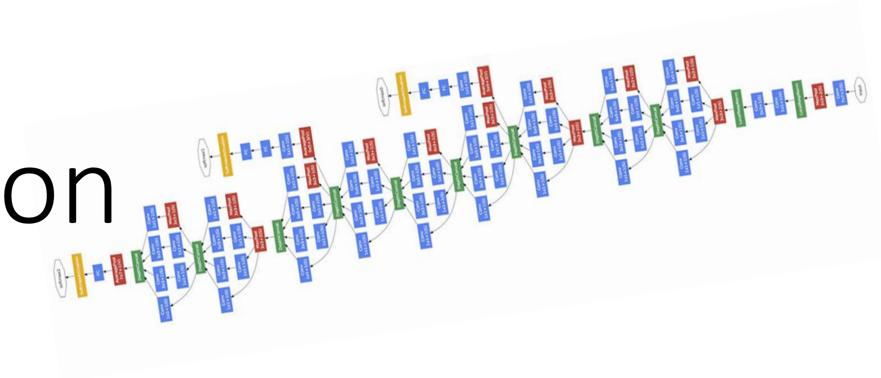
⋮

$$P(y) = \frac{1}{N}\sum_n P(y^n|x^n)$$

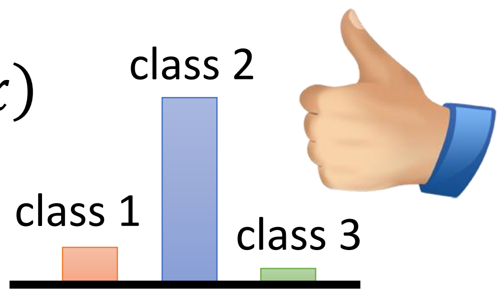


Uniform distribution means higher variety

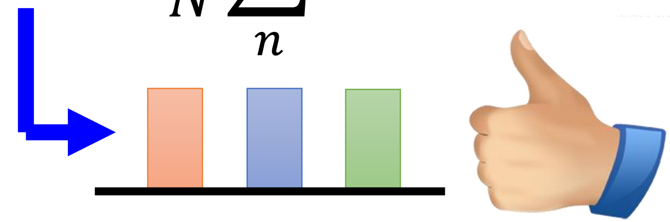
Objective Evaluation



$P(y|x)$



$$P(y) = \frac{1}{N}\sum_n P(y^n|x^n)$$



Inception Score

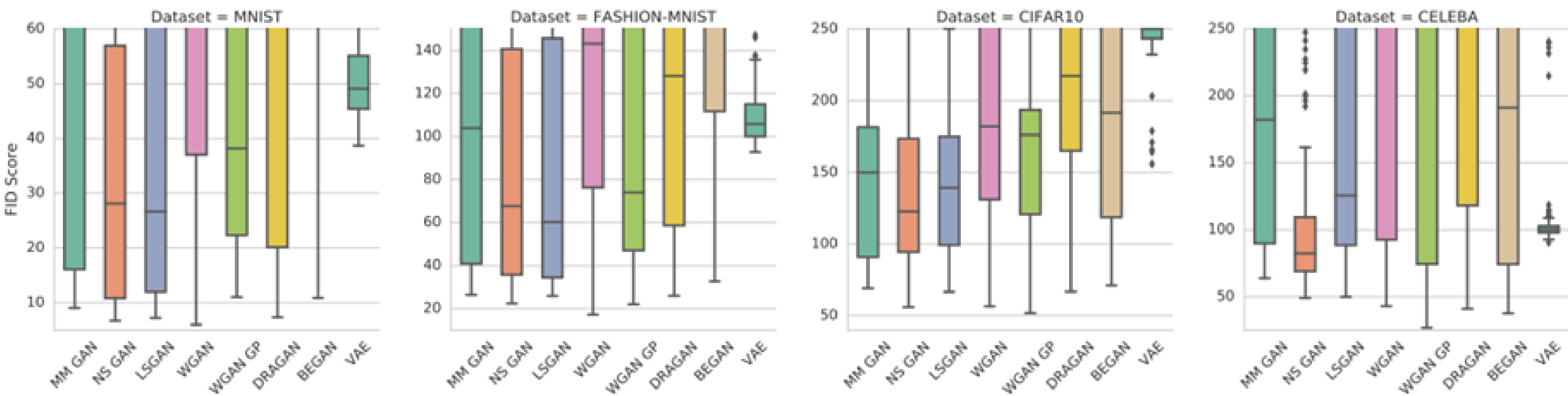
$$= \sum_x \sum_y P(y|x) log P(y|x)$$

Negative entropy of P(y|x)

$$- \sum_y P(y) log P(y)$$

Entropy of P(y)

GAN	Discriminator Loss	Generator Loss
MM GAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] + \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{GAN}} = -\mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}}$
NS GAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{NSGAN}} = \mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}}$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{NSGAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\log(D(\hat{x}))]$
WGAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}} = -\mathbb{E}_{x\sim p_d}[D(x)] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{WGAN}} - = \mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}}$
WGAN GP	$\mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}} = \mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_g}[(\|\|\nabla D(\alpha x + (1-\alpha\hat{x})\|\|_2 - 1)^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{WGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
LS GAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{LSGAN}} = -\mathbb{E}_{x\sim p_d}[(D(x)-1)^2] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{LSGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[(D(\hat{x}-1)^2]$
DRAGAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{DRAGAN}} = \mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_d+\mathcal{N}(0,c)}[(\|\|\nabla D(\hat{x})\|\|_2 - 1)^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{DRAGAN}} = -\mathcal{L}_{\mathrm{D}}^{\mathrm{NS\ GAN}}$
BEGAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{BEGAN}} = \mathbb{E}_{x\sim p_d}[\|\|x-\mathrm{AE}(x)\|\|_1] - k_t\mathbb{E}_{\hat{x}\sim p_g}[\|\|\hat{x}-\mathrm{AE}(\hat{x})\|\|_1]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{BEGAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\|\|\hat{x}-\mathrm{AE}(\hat{x})\|\|_1]$



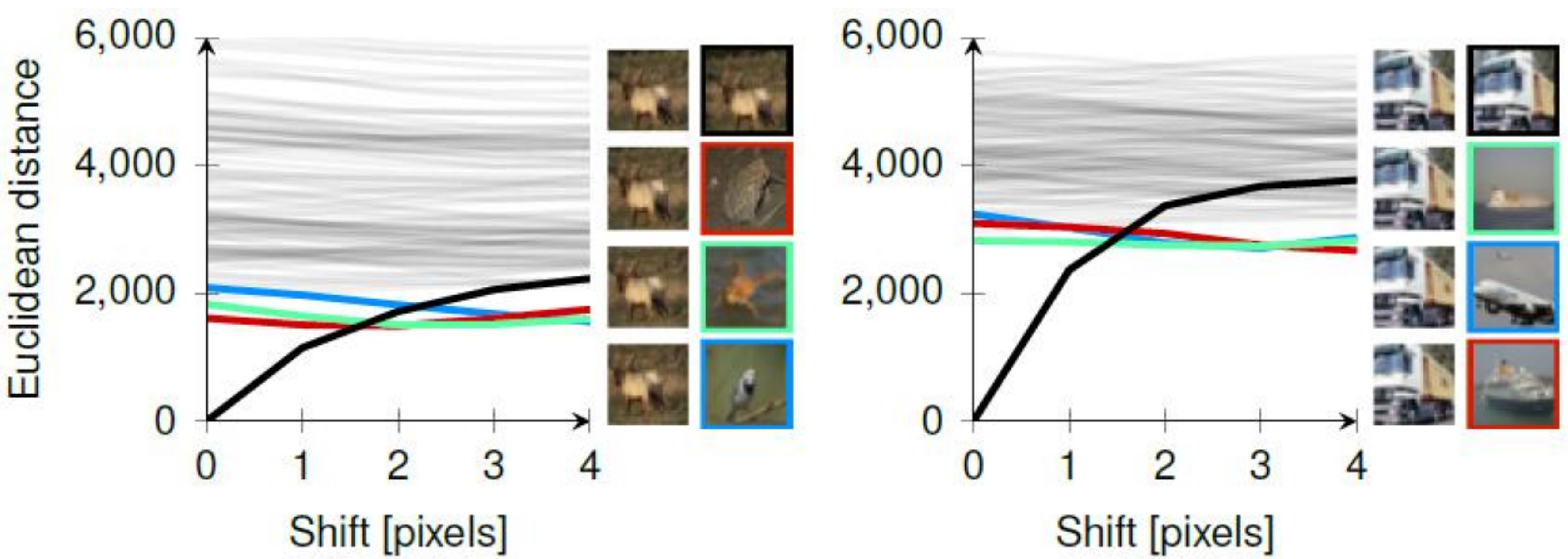
Smaller is better

FIT: https://arxiv.org/pdf/1706.08500.pdf

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, Olivier Bousquet, "Are GANs Created Equal? A Large-Scale Study", arXiv, 2017 https://arxiv.org/pdf/1511.01844.pdf

We don’t want memory GAN.

- Using k-nearest neighbor to check whether the generator generates new objects



Missing Mode ?

Mode collapse is easy to detect.

?