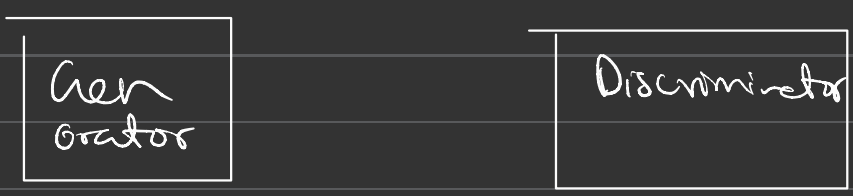
General Adversial Networks

Noise & class $\xi, Y \rightarrow X$ Features

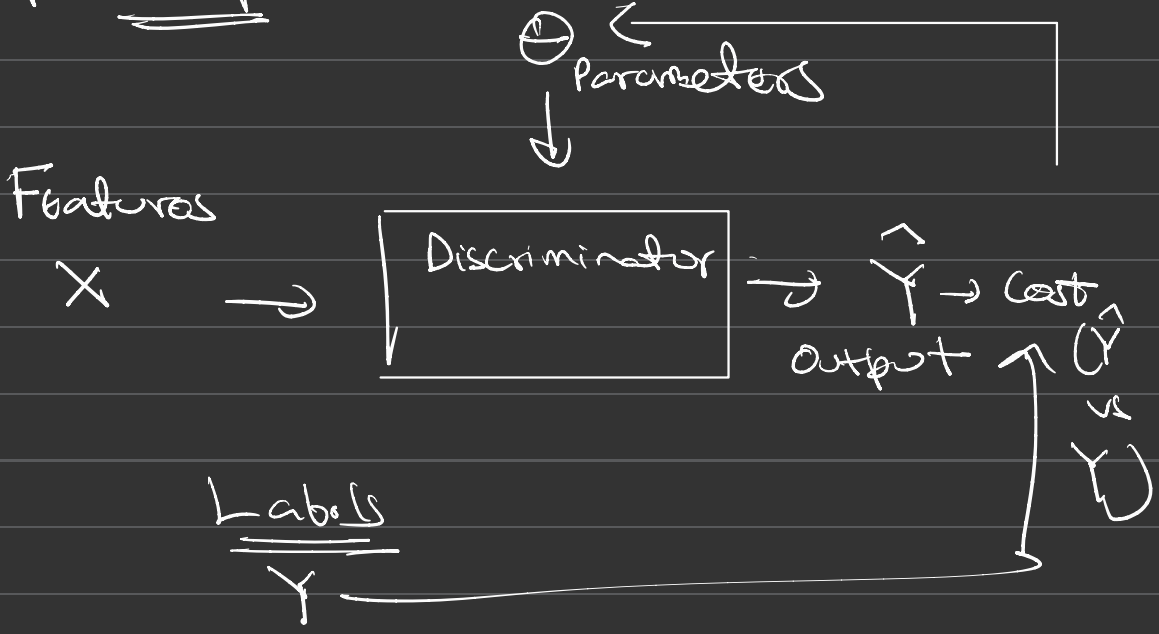
$P(X|Y)$



↳ Similar to decoder in VAE

→ Later, we don't need the discriminator and we can use the generator.

☆ Classifiers



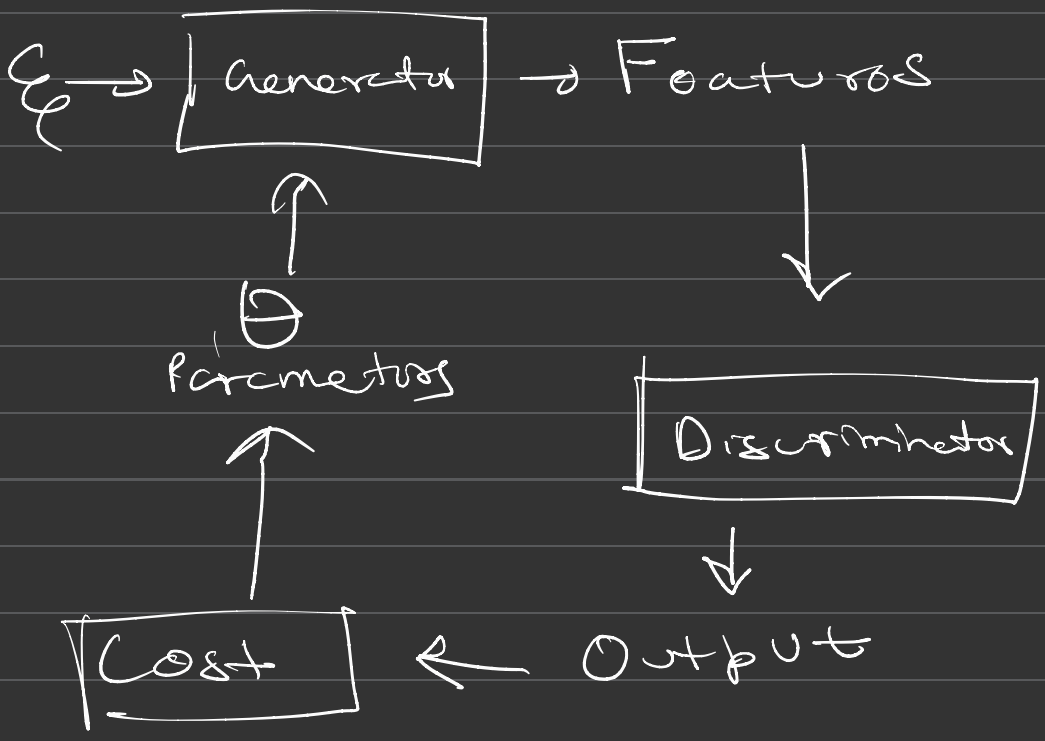
↗ Discriminator

$P(Y|X)$ — Y: Class, X: Features

↗ Generator

$P(X|Y)$ — X: Features, Y: Class

Generator



☆ Binary Cross Entropy

Average loss of the whole batch

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \log h(x^{(i)};\theta) + (1-y^{(i)}) \log (1-h(x^{(i)};\theta)) \right]$$

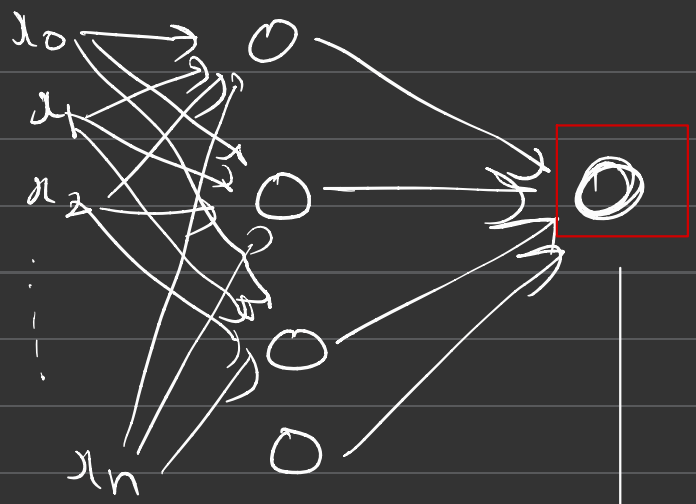
$h(x^{(i)};\theta)$: Prediction

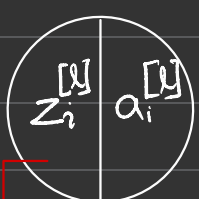
$y^{(i)}$: Label

$x^{(i)}$: Features

θ: Parameters

→ We can't use a "supervised" Generator with a normal discriminator and vice-versa





$$z_i^{[l]} = \sum_{i=0}^{n} w_i^{[l]} a_i^{[l-1]}$$

$$a_i^{[l]} = g^{[l]}(z_i^{[l]})$$

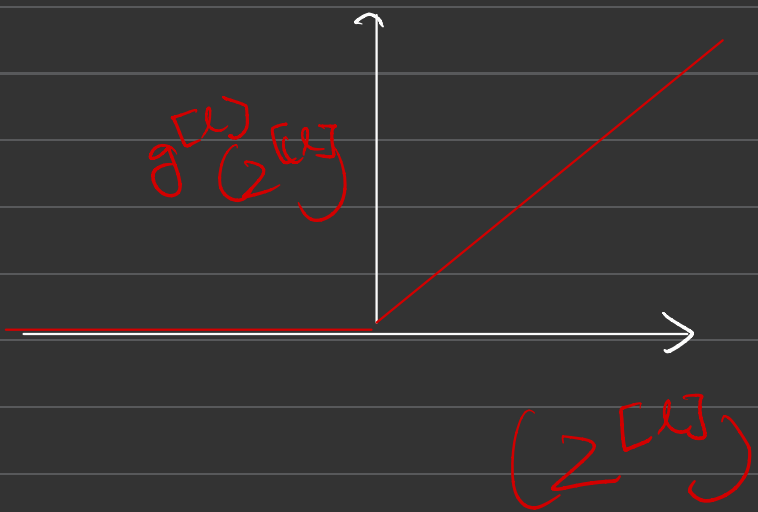
Diff → for backprob.

Non-linear → for approx. complex function

activation function

(diff and Non-linear)

★ ReLU

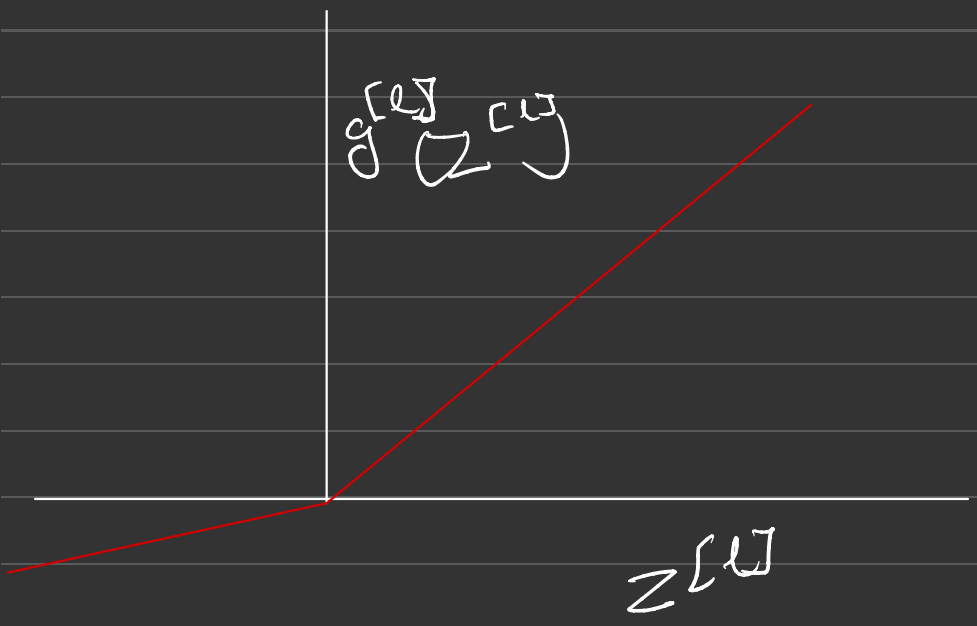


$$g^{[l]}(z^{[l]}) = \max(0, z^{[l]})$$

★ Dying ReLU → (due to the fact that nets not accomodating to the change)

↳ solution: Leaky ReLU

☆ Leaky ReLU

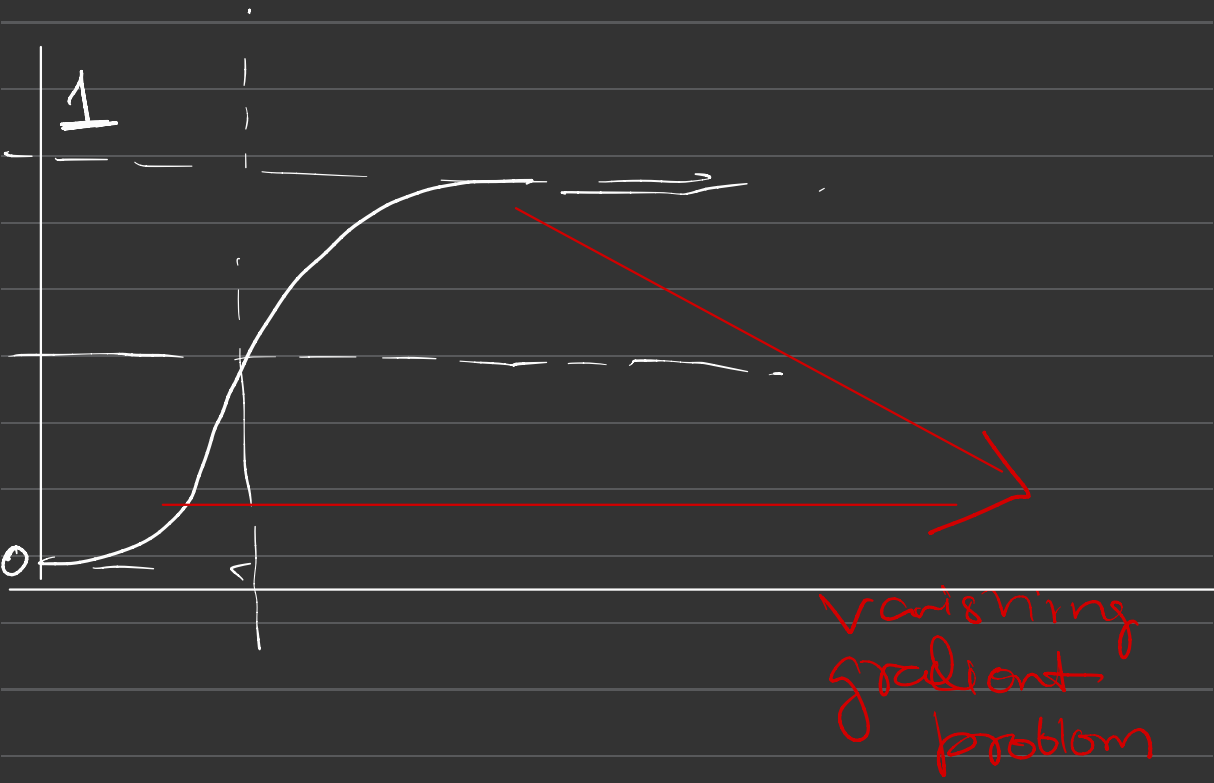


$$g^{[l]}(z^{[l]}) = \max(az^{[l]}, z^{[l]})$$

Solves the dying ReLU problem

Value often set to 0.1

★ Sigmoid activation



★ Batch Normalisation

→ to counter covariate shift

→ To create similar distribution of features of factors that will result in a smoother cost function

→ Speeds up the learning process

★ Batch normalisation : training

$$z_i^{[l]} = \sum_{i=0} w_i^{[l]} a_i^{[l-1]}$$

$$\hat{z}_i^{[l]} = \frac{z_i^{[l]} - \mu_{z_i}^{[l]}}{\sqrt{\sigma_{z_i}^{2\,[l]} + \epsilon}}$$

mean

variance

$\epsilon \to$ to avoid denom $\neq 0$

$$y_i^{[l]} = \gamma \hat{z}_i^{[l]} + \beta$$

γ: Scale Factor

β: Shift Factor

★ Batch Normalization : Toss

$$\hat{z}_i^{[l]} = \frac{z_i^{[l]} - E(z_i^{[l]})}{\sqrt{Var(z_i^{[l]}) + \epsilon}}$$

(now these are fixed values)

☆ Convolution

Stride → how many pixels you move in one time with the filter
(no. of steps basically)

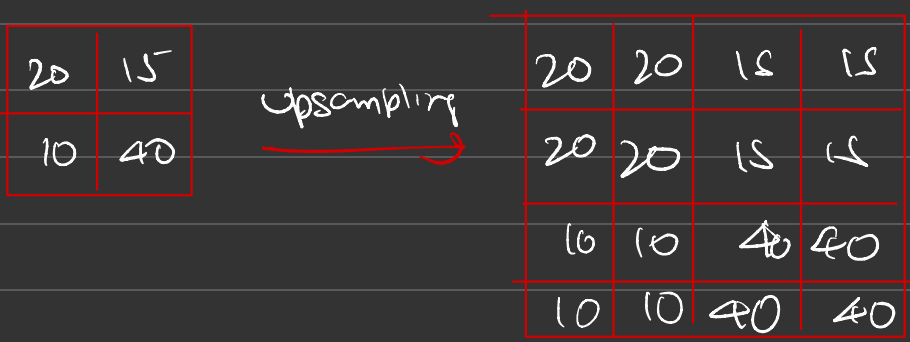
Padding → extra frame of pixels used to pay attention to all the parts of the images

Pooling → To reduce the size of the images

⭐ Upsampling

↳ nearest neighbours

↓



→ Pooling & upsampling do not have any learnable parameters.

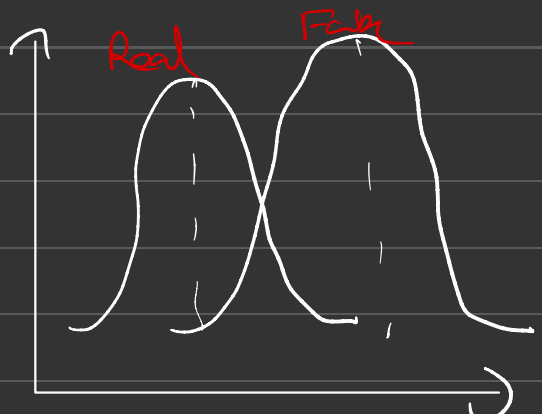
↳ But transposed convolutions do!!

★ Mode collapse

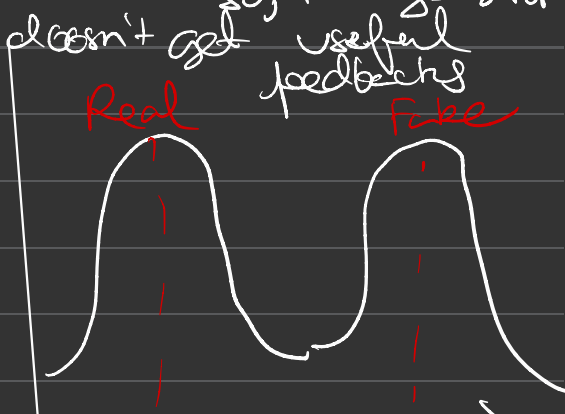
→ Any peak on the density function is a mode

→ Mode collapse happens when the generator gets stuck in one mode

★ Problem with BCE

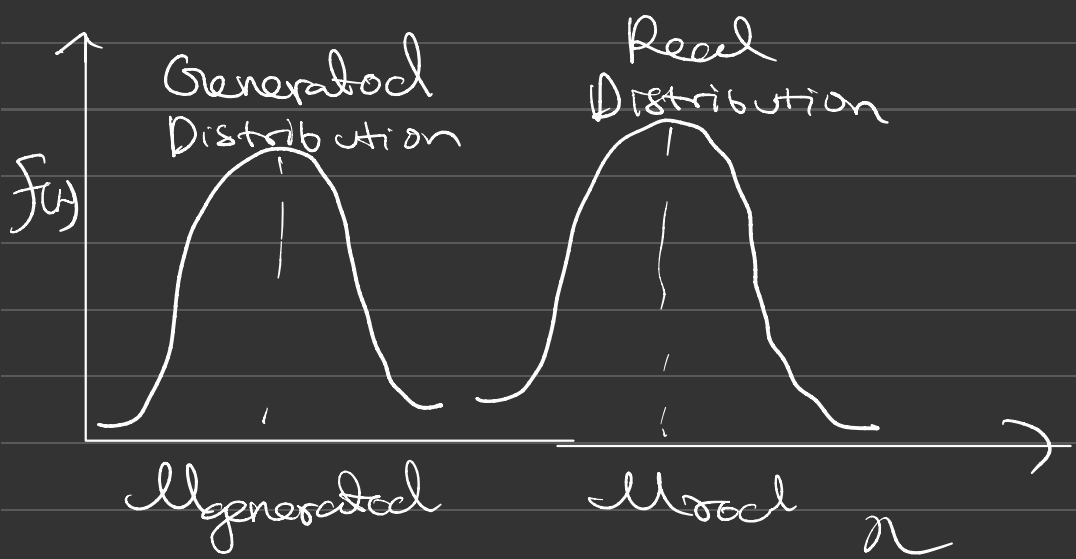


Discriminator in the beginning



discriminator later

★ Earth Mover's Distance



→ It calculates the "Effort" to make the generated distribution equal to the real distribution

→ Solves gradient vanishing problem

☆ Wasserstein Loss

$$\min_{g} \max_{c} E(c(x)) - E(c(g(z)))$$

g: generator, c: critic

BCE	W-Loss
output $0 < \alpha \leq 1$	output any number

→ also ndp with mode collapse

✱ Condition of W-Loss

Critic should be 1-Lipschitz continuous

The norm of the gradient should be at-most 1 for every point

✱ 1-Lipschitz cont.

$$\| \nabla f(w) \|_2 \leq 1$$

How to ensure it?

→ Weight clipping forces the weights of the critic to a fixed interval

Gradient descent to update weights

↓

Clip the critic's weight

(Limits the learning ability of the critic)

(better method)

★ Gradient Penalty

W-loss Function ↑

$$\left[\min_{g} \max_{c} E(c(x)) - E(c(g(z)))\right] + \lambda \text{reg}$$

↓

regularisation of the critic's gradient

Final form of W-Loss → the main Function

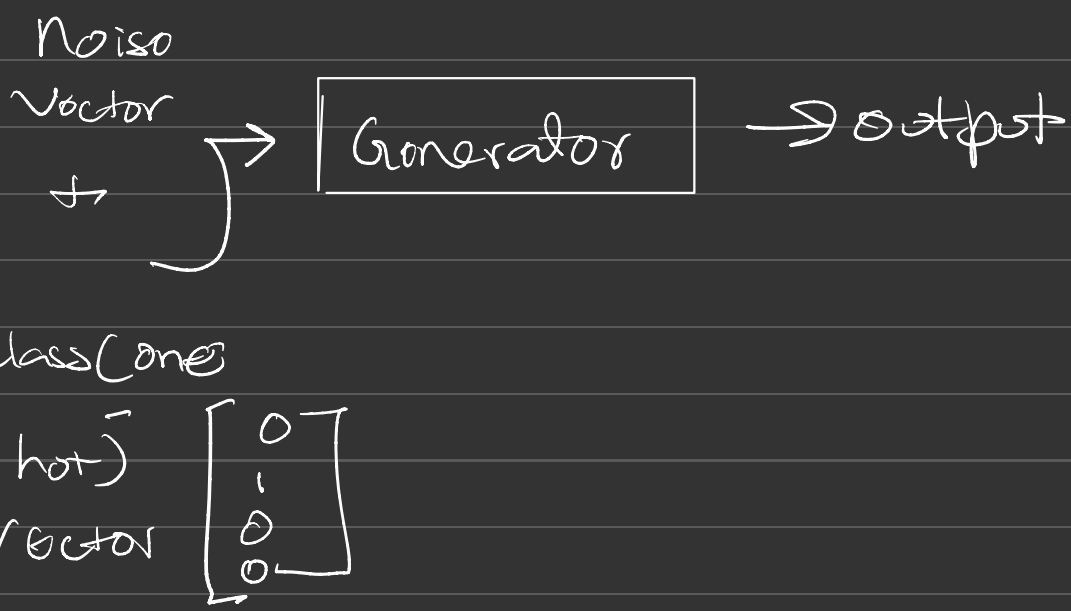
$$\min_{g} \max_{c} E(c(x)) - E(c(g(z))) + \lambda E(\|\nabla c(\hat{x})\|_2 - 1)^2$$

→ makes GAN less prone to mode collapse and vanishing gradient

→ tries to make the critic 1-L (cont. and diff.

* Conditional Generation

→ Training dataset needs to be labeled.



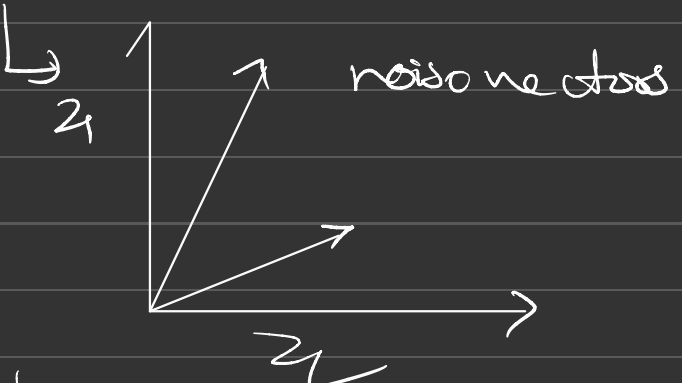
★ Controllable GAN

↳ Control by changing the noise vector

→ does not need a labeled training dataset

→ Z-space

↳



↳ Problem:

↳ Z-space Entanglement

★ Classifier Gradients