Discriminative Models — $\mathbb{P}\{Y|X\}$

Generative Models — $\mathbb{P}\{X|Y\}$

$\xi, y \longrightarrow X$

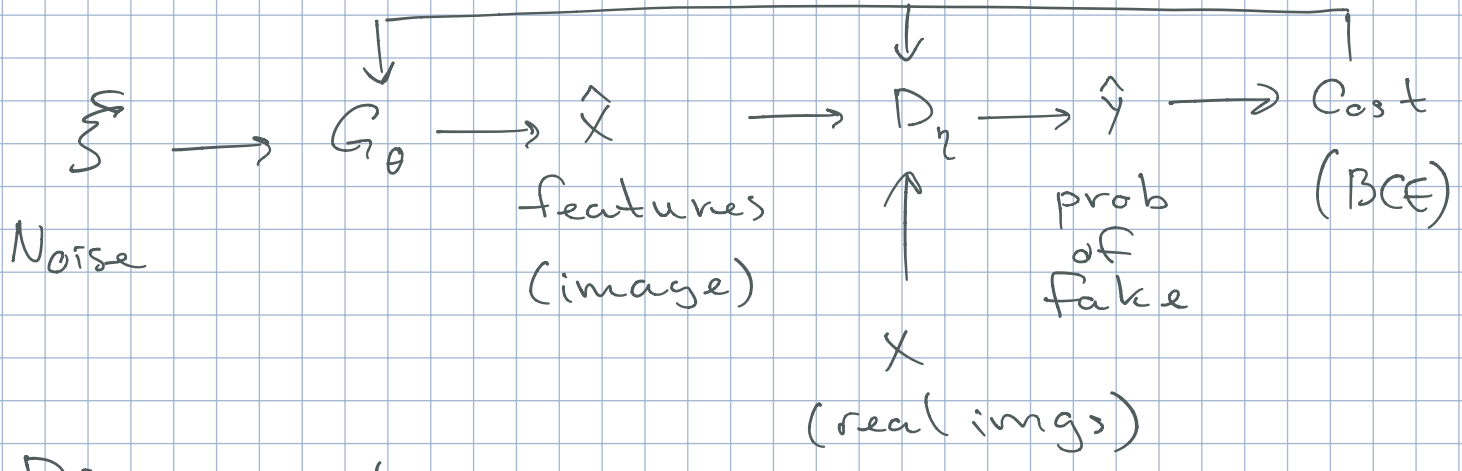
ξ: noise, y: class

Examples : VAE, GAN

GANs Intuition

Generator

Learns to make fakes that look real



Discriminator

Learns to distinguish real from fake

Binary classifier : $Y = \{\text{real}, \text{fake}\}$

$\mathbb{P}\{Y|X\}$ = prob of being fake

This prob. will be given back to generator to improve its efforts

Training

* Discriminator

  We use $\hat{X}$ and X, compute BCE and update D_η

* Generator

  We use only $\hat{X}$ (new), compute BCE and update G_θ

So we alternate between training D_η and G_θ

Remark

Models should improve together so we should start training with G_θ and D_η of comparable "skill" levels.

Common Issue: Superior D_η

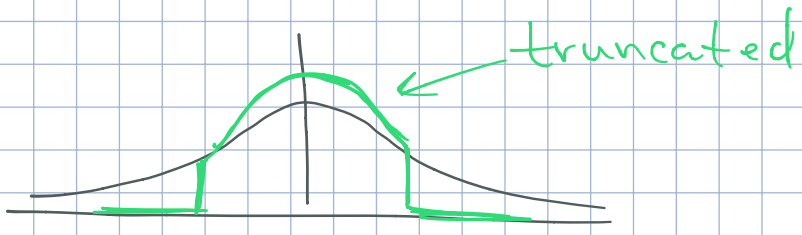
When D_η classifies all $\hat{X}$ as 100% fake then G_θ doesn't know how to improve

Truncation Trick

To generate $\hat{x}$ with G_θ we use

$$\xi \sim \mathcal{N}(0, I)$$

Observation: while training G_θ it deals with noise vectors close to zero a lot more often than with noise vectors from tails

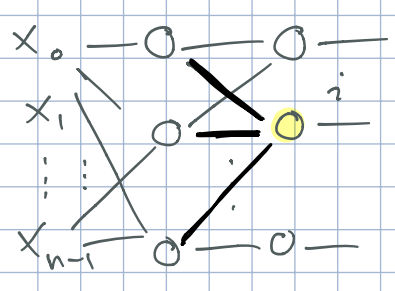


We can make a tradeoff between Quality and Diversity of $\hat{x}$ by truncating distr. from which we sample noise vectors.

Batch Normalization

Used to speed up training

Procedure



Lets look at

$$z_i^l = \sum_j a_j^{l-1} w_{ji}^l$$

Given batch of k logits

$$z_{i,1}^l, \dots, z_{i,k}^l$$

We normalize z_i^l using batch statistics

$$\hat{z}_i^l = \frac{z_i^l - \mu}{\sqrt{\sigma^2 + \varepsilon}},$$

where $\mu = \frac{1}{k}\sum_j z_{i,j}^l$ and $\sigma^2 = \mathrm{Var}(z_{ij}^l)$

But, we not only normalize them:

$$\hat{z}_i^l \leftarrow \gamma z_i^l + \beta,$$

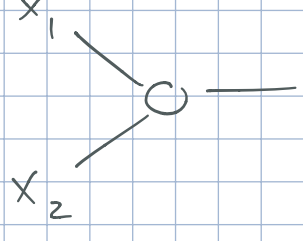
where γ and β – learnable parameters of BatchNorm layer.

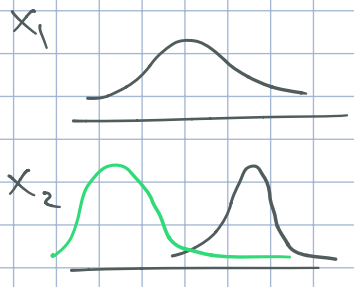
Finaly, $a_i^l \leftarrow \mathrm{activation}(\hat{z}_i^l)$

During evaluation, instead of using batch stat, we use μ and σ^2 computed over all training set

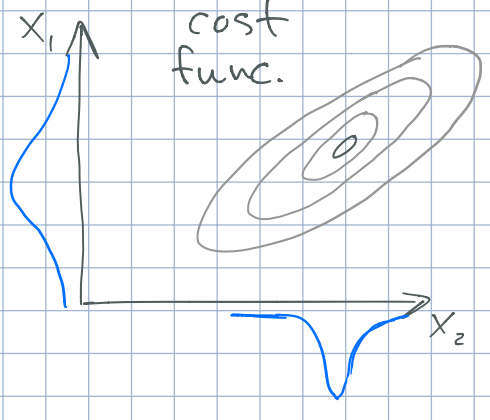
Intuition

Reduce internal covariate shift.





Values Distribution



If later in time distribution of x_2 will change but labels will be the same, then cost function landscape might change as well, therefore we will not be in optimal point.

However, if $x_i \sim N(0,1)$ and we will normalize x_i later then cost function landscape will not change.

Another observation is that when x_i distributions are different then same change for w_i might have very different impact on learning.

Transposed Convolutions

Recall that Pooling layers are used to reduce dimensionality.
But what if we want to do the opposite?

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} = \begin{bmatrix} aw_1 & aw_2 + bw_1 & bw_2 \\ aw_3 + cw_1 & aw_4 + bw_3 + cw_2 + dw_1 & bw_4 + dw_2 \\ cw_3 & cw_4 + dw_3 & dw_4 \end{bmatrix}$$

Learnable params.

Mode Collapse

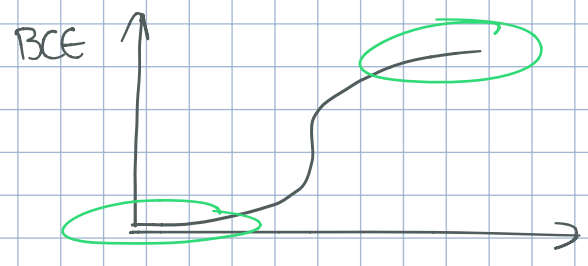
G_θ samples from latent distribution which usually has multiple modes.

However, if discriminator is very good at classifing object from all modes except one then it will force generator to produce object only from this particular mode — end of training

This happens because of BCE loss

Problem with BCE loss

Since D_η solves simpler task it could learn faster than G_θ, and if so it will output values close to 1 and 0 therefore



BCE will be at values with close to zero gradient ⇒ No useful information to imp. G_θ.

↓

Vanishing Gradients.

Earth Mover's Distance

Intuition: G_θ in some sense wants to minimize distance between distributions of generated features and real features.

EMD Measures how much effort it's needed to move one distribution to another

EMD does not suffer from vanishing gradients problem.

Wasserstein Loss

Approximation of EMD

Recall BCE

$$J = -\frac{1}{m}\sum y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)$$

$$y_i = \begin{cases} 1, & \text{if } x_i - \text{real image} \\ 0, & \text{otherwise} \end{cases}$$

$$-\frac{1}{m}\sum y_i \log D_\eta(x_i) + (1-y_i)\log\left(1 - D_\eta(G_\theta(z_i))\right)$$

$$= -\mathbb{E}_{x\sim p}\log D_\eta(x) - \mathbb{E}_{z\sim r}\log(1 - D_\eta(G_\theta(z)))$$

$$\Rightarrow \max_\theta \min_\eta -\mathbb{E}\log D(x) - \mathbb{E}\log(1 - D(G(z)))$$

because generator wants to be good at fooling discrim.

because discriminator wants to be good at discriminating between reals and fakes

W-loss : $\min_g \max_c \mathbb{E}\, c(x) - \mathbb{E}\, c(g(z))$

critic (discriminator but with output not limited by range $[0,1]$)

wants to push distributions of fakes and reals apart

generator wants to get them as close as possible.

Remark

Since output of critic is not bounded, it does not suffer from vanishing gradients problem

Continuity Condition for Critic

c should be 1-Lipschitz Continuous, i.e.

$\forall x \quad \left\| \frac{\partial c(x)}{\partial x} \right\|_2 \leq 1$

Needed for stable learning

Q: How to make sure that critic is 1-L cont.?

A:

* Wheight clipping (of critic during training)
  (Much hyperopt needed)

* Gradient penalty

$$\min_g \max_c \mathbb{E}\, c(x) - \mathbb{E}\, c(g(z)) + \lambda \cdot \text{reg} \;,$$

where $\text{reg} := \left(\|\nabla_{\hat{x}} c(\hat{x})\|_2 - 1 \right)^2$, $\hat{x} = \varepsilon x + (1-\varepsilon) g(z)$

Intuition is that we take some point $\hat{x}$ on the path from x to $g(z)$ and try to enforce that gradient of critic in that point is equal to 1.

Soft condition to 1-L continuous.

Remark

Works good in practice.

Conditional GAN & Controllable Generation

Conditional Generation Intuition

Allows to choose, for example, class of object that will be generated, in contrast to what we have seen so far.

Therefore, training dataset needs classes labels

Inputs Modification

Q: How to add information about a class to G_θ and D_η?

* Generator: Concat noise vector with one-hot class vector.

But how G will learn to generate objects of selected class?
Because of the feed back from D.

* Discriminator:

$$\left.\begin{matrix} x \\ y \end{matrix}\right\rangle D_\eta \rightarrow \mathbb{P}\{x \text{ is real} \mid y\}$$

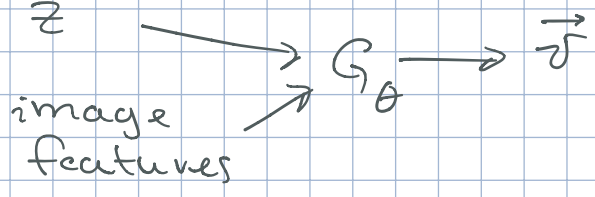
because D outputs prob wrt class y
then it will force G to generate objects from specefied class eventualy.

! Since x - multichannel input, info about y could be added using channels as well — append $|C|$ additional channels to x with only one of them with ones.

Multimodal Example

Generation of tags for images

* Pretrained net to extract images features
* Pretrained LM to map tags to vectors
* Generator

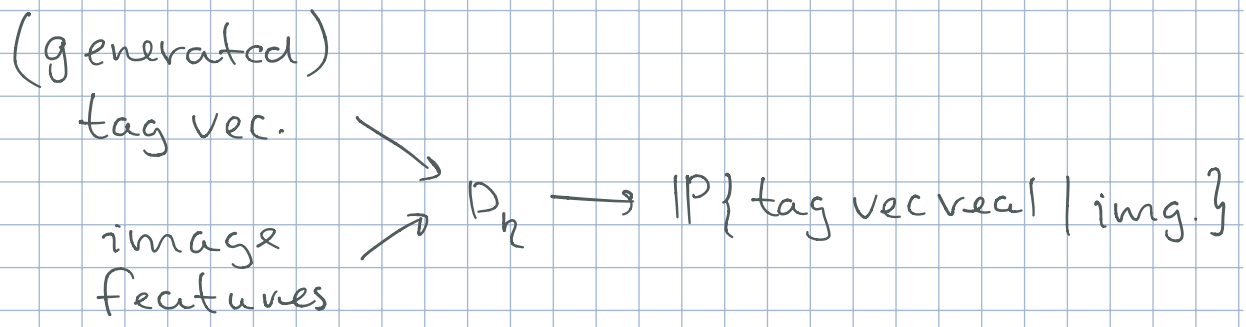


find $N(\vec{v})$ – nearest neighbours of $\vec{v}$ among vecs. of tags.

$|N(\vec{v})| = k$

Run multiple times, select most frequent words

* Discriminator



Controllable Generation

Allows to control features of generated objects (even after model was trained)

For example, if G generates faces, features - Age, Gender, Eyeglasses, ---.

Intuition: tweak noise vector z

Therefore, in contrast to Conditional Gen. it is not required to have labeled data.

Main Idea

$\forall$ z - noise vector to which corresponds

G(z) - image of woman with dark hair.

Goal is to find z' s.t. G(z') - img of the same woman but with light hair.

Thus, $z - z' =: d$ - direction for hair color feature.

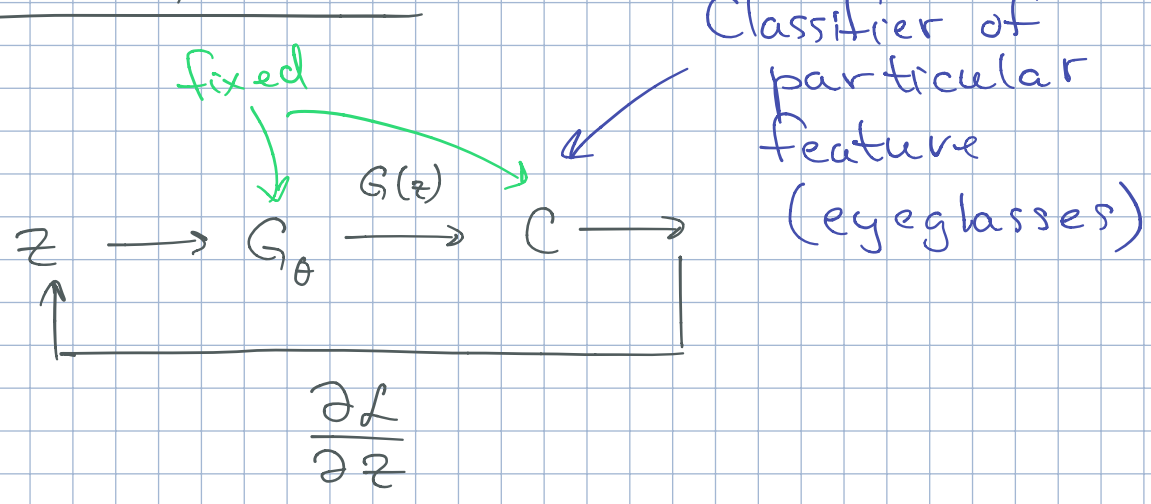
Given another img of person with dark hair, we can add d to its noise vec to get the same person with lighter hair color.

Q: How can we find directions corresponding to different features?

Problem: z-space entanglement

basicly, when while changing one feature we change some other features as well.

A: Classifier Gradients



We want to find d (i.e. z' with particular feature)

We start with z without this feature and will change it

$$z \to z_1 \to z_2 \to \cdots \to z_n$$

while C will not tell us that z_n has a feature.

$$z_{i+1} \leftarrow z_i - \eta \frac{\partial BCE}{\partial z_i}$$

i.e. we want to minimize BCE by tweaking z

Remark in the given scenario

$$BCE = 1 \cdot \log C(G(z)) + 0 \cdot \log(1 - C(G(z)))$$

therefore $\min_z BCE \iff \max_z C(G(z))$

thus

$$z_{i+1} \leftarrow z_i + \eta \frac{\partial C}{\partial z_i}$$