Deep Unsupervised Learning

L2 - Autoregressive models

Problems to solve with likelihood based models.

- generating data
- Compressing data
- anomaly detection

- learning a distri P, - computing $p(x)$ for x
  - sampling $x \sim p(x)$

- estimate distri of complex, high dim data
- computational and statistical efficiency.
  - Efficient training & model rep
  - Expressiveness and generalisation
  - Sampling quality & speed
  - Compression rate & speed.

Histograms

estimate part of underlying data distri from samples.

- histogram plots on occurance of each data point on sampling
- Inference -> lookup.
- Sampling -> sample from uniform distri, then move through the space from that point by probability mass that point, then pick it.
- failure in high dim.
- won't generalize for test cases.

Sathvik Vaduri

Parameterised Distri:

- fit a param distri over training data
- use for approx.

$$P_\theta(x) \approx P_{data}(x)$$

- Solve using MLE, SGD
- P_θ will be a dnn.
- Probability distri over x should satisfy $\sum_x P_\theta(x) \geq 1$ So we keep it >1.

Autoregressive models

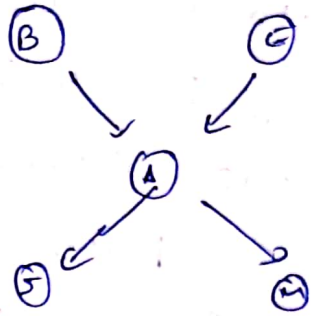
Bayes net struct

chain rule

$P(B, E, A, J, M)$

$= P(B) \cdot P(E|B) \cdot P(A|E,B) \cdot P(J|A,E,B) \cdot P(M|J,A,E,B)$

$P(M|A)$



→ assumptions

The sparsity from assumptions leads to smaller tables for the conditionals. This makes it representable.

Autoregressive models parameterise these conditionals.

Bayes net

$$\log p_\theta(x) = \sum_i^d \log p_\theta(x_i | \text{parents}(x_i))$$

↓

for an image, every pixel is an entry in x, each colour channel is an entry.

$$\log p(x) = \sum_i^d \log p(x_i | x_{1:i-1})$$

this is going to be parameterised

$(x_1, x_2) \in x$

model: $P(x_1, x_2) = P(x_1)\, P(x_2|x_1)$

$P(x_1)$ ↓ histogram

$P(x_2|x_1)$ ↓ parameterize using MLP, o/p on softmax

For High dims

- order of d params $(O(d))$
- would have been $O(\exp(d))$ in tabular

issues:

- for problems like text gen, d is very large
- No info sharing b/w conditionals, limited generalisation.

Soln:

- RNNs,
- Masking

RNN autoregressive models: (char-rnn)

$$\log P(x) = \sum_{1}^{d} \log p(x_i | x_{1:i-1})$$

x: sequence of chars

x_i: character at i^{th} position.

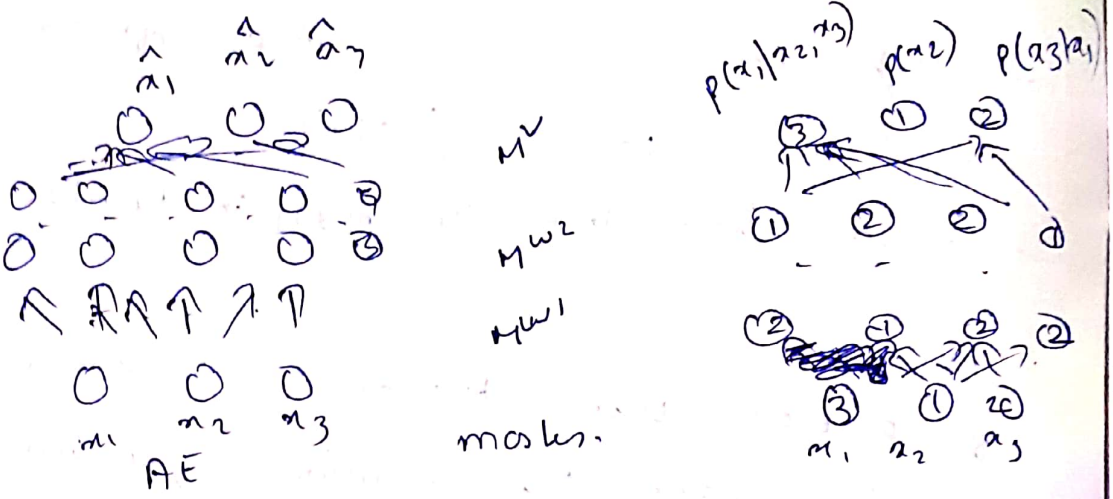
- can also generate mnist using rnn this way.

Masking based models.

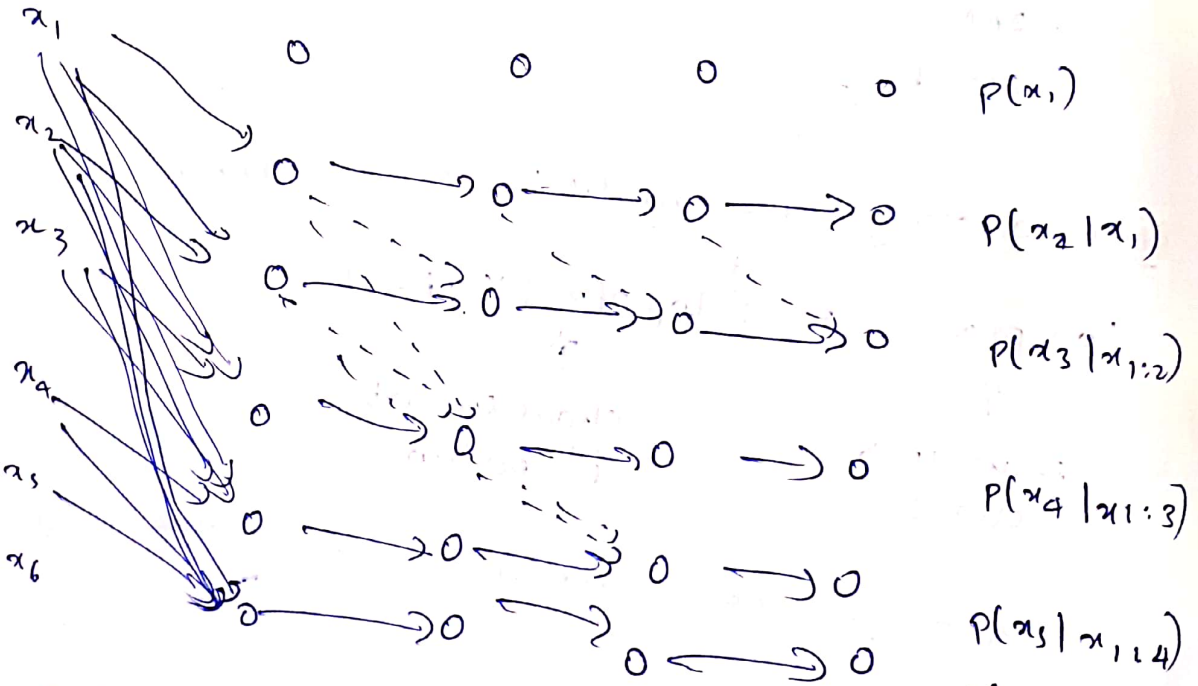
- parallised computation of all conditionals
- Masked MLP
- Masked conv & self-attn.

Masked autoencoder for distribution estimation (MADE)

Autoencoder × Masks → MADE

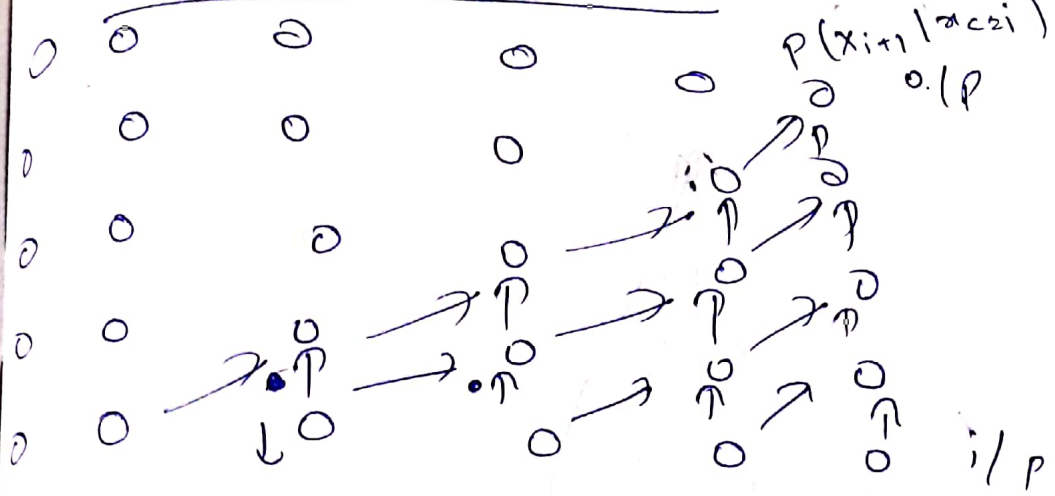


- P conditional probs can be obtained by removing a few edges.
- Done by defining which edges go where.



- All the dotted lines can be kept.
- Orderings matter.

Masked 1D Convolution



- thus ops can be masked conv, same operation everywhere.
- constant param count for variable length
- conv highly hardware optimised
- But limited receptive field

Wavenet

- Improved receptive field, dilated conv, with exponential, dilation
- gated Res blocks, skip connection
- works on mnist with position encoding.

Pixel CNN

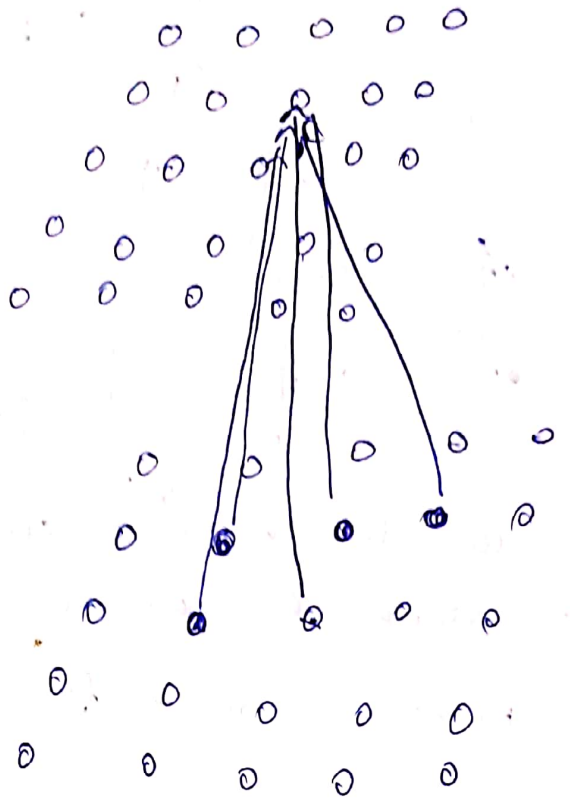
- Masked 2D Conv
- change the filter, mask it directly.

$x_1 \ldots x_n$ → raster scan order.

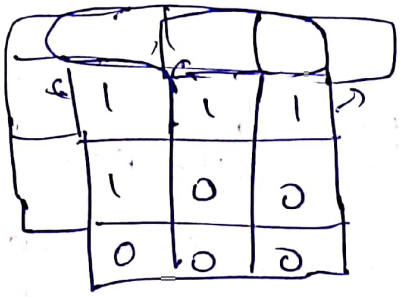
x_i

- mask →
$$\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}$$

- multiple layer



receptive field of 2 layer

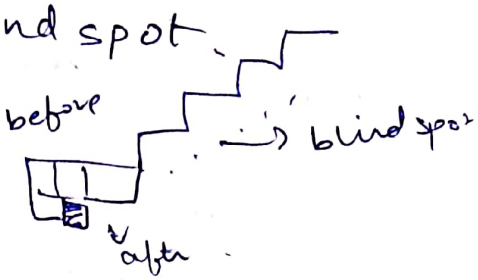


this will keep growing.

∴ it grows such that the receptive field shifts to things before, nothing on itself or what comes after.

Keep doing pixel after pixel,

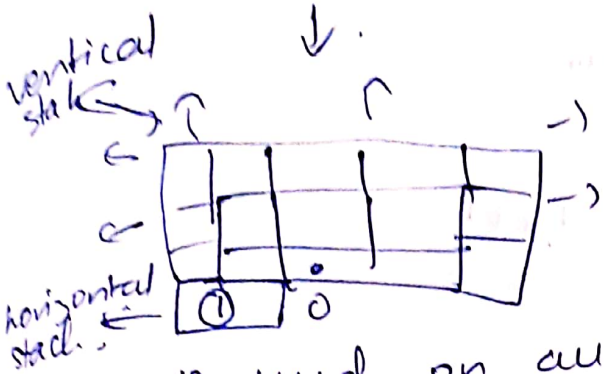
- But there is a blind spot.



To avoid the blindspot, change kernel to

o/p -> softmax over 256

receptive field



this is used on all pixels.

But in last layer the centred point is trained again to predict the next one. (would be cheating otherwise as it saw the point before)

But now there is a blindspot at (1)

a 1D filter is used for that.

gated PixelCNN

$$y = \tanh(W_{k,f} * x) \odot \sigma(W_{k,g} * x)$$

PixelCNN++

- nearby pixels are likely to co-occur.

$$p(x \mid \pi, \mu, s) = \sum_{i}^{K} \pi_i \left[\sigma((x+0.5-\mu_i)/s_i) - \sigma((x-0.5-\mu_i)/s_i)\right]$$

μ -> mean
s -> slope

parametrising sigmoid
- cummulative distri

- The diff b/w 2 sigmoids give density
- Capture long dependencies efficiently by downsampling (skip connection)

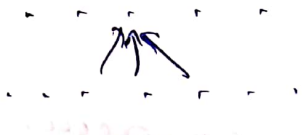
Masked Attention

- Multiheaded self attn.
- good masking
- unlimited receptive field

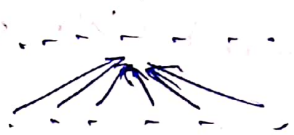
Attention

query, a hidden vector, typically generated by a previous layer; Another part of n/w will have key, value pairs. The query will do inner products with keys. Then weighted sum of values based on how good of match b/w key and query.

$$A(q, k, v) = \sum_i \frac{e^{q k_i}}{\sum_j e^{q k_j}} v_i$$



conv



attn.

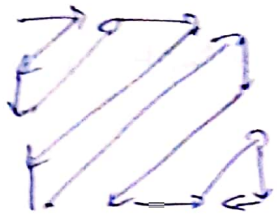
masking unnecessary connections

- subtract a very -ve number

$$\sum_i \frac{e^{q k_i - \text{masked}(k_i, q) \cdot 10^{10}}}{\sum_j e^{q k_i - \text{masked}(k_i, q) \cdot 10^{10}}} v_i$$

∴

arbitary ordering.



zigzag ordering.

- lot of param sharing still happening

PixelSNAIL:

Masked Attn + Conv.

Class conditional PixelCNN

- One-hot encoding of label
- force label image to be generated

Disadvantage of masked autoregressive models

- Sampling each pixel: 1 forward pass.
- 11 mins to generate 16 32 x 32 on GPU

how to overcome?

- breaking auto regressive pattern.
- make groups and do parallelize
- Speedup by caching activations

How to get latent rep of Pixel CNN?

- use fisher score

$$L(x; \theta) = \nabla_\theta \log p_\theta(x)$$

L3- flow models

- mixture of gaussians (Refer DeepBayes)
- Practical Param of flows.
  - Cumulative Densities for
    - eg: mix of gaussian.
  - NN each layer is a flow, then sequencing of layer = flow
    - Can't use ReLU as we can't be inverted.
    - sigmoid can be used.
    - tanh not a good idea as depends on values to fit data

2D flows

$x_1 \rightarrow z_1 = b_\theta(x_1)$

$x_2 \rightarrow z_2 = b_\phi(x_1, x_2)$

x_1, x_2 are not independent

training obj:

$$\max_\theta \sum \log P_{z_1} \log(b_\theta(x_1)) + \log\left|\frac{dz_1}{dx_1}\right|$$

$$+ \log P_{z_2} \log(b_\phi(x_1, x_2)) + \log\left|\frac{dz_2}{dx_2}\right|$$

N-D flows

for high d data

Images $\xrightarrow{\text{inverse } f}$ distri

$\xleftarrow{f^{-1}}$ Sampling

N-D using Autoregressive flows.

$z_1 \sim P_\theta(x_1)$

$x_1 = f_\theta^{-1}(z_1)$

$x_2 \sim p_\theta(x_2|x_1)$ $\quad x_2 = f_\theta^{-1}(x_2; x_1)$

- sampling is an invertible mapping
- lot of sampling for large images.

Inverse AR flow

$x \to z$

$z_1 = f_\theta^{-1}(x_1) \quad x_1 = f_\theta(z_1)$

$z_2 = f_\theta^{-1}(x_2; z_1)$

- IAF Sampling is fast

- Both AF & IAF is dup as no of valid soln. param sharing
- again train with MLE

NICE / Real NVP

splitting variables in half

$z_{1:d/2} = x_{1:d/2}$

$z_{d:2:d} = x_{d/2:d} \cdot s_\theta(x_{1:d/2}) + t_\theta(x_{1:d/2})$

s_θ & t_θ can be NNs with no restriction

Papers to look at:

glow, flow++, ffjord.

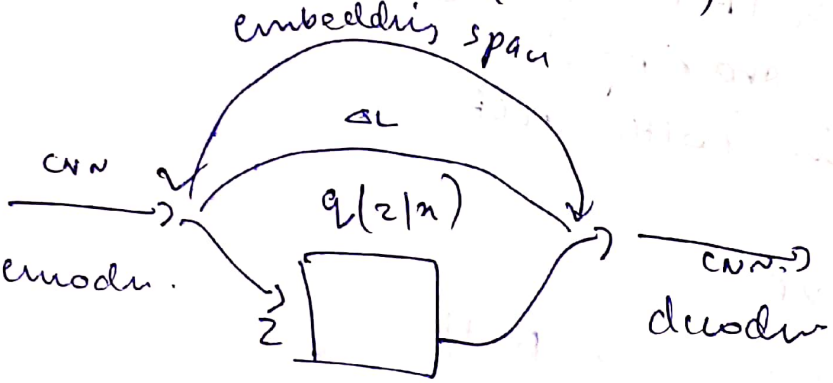
Latent Variable Models (contd)

- AR models are slow to sample as all pixels are assumed dependent
- make some part of obs space independent on some LV

- Train & opti → backprop

additional work on VAEs

VQ-VAE (vector quantized)



↳ discrete latent space

[Read full paper]

VQ VAE 2.0

- two levels of VAE on encoder decoder
- hierarchical latent space

GANs

original gan.

- loss

$$\min_G \max_D \; E_{x \sim p_{data}}[\log D(x)] + E_{z \sim p(z)}[\log(1 - D(G(z)))]$$

↓ classify. ↓ generate

Inception score for evaluation

- take pre trained Inception v3 on IM.

$p(y|x)$.

- Marginal label distri $P(y) = \int_x p(y|x) p_g(x)$

- $IS = \exp\left(E_{x \sim p_g}[D_{KL}[p(y|x) || p(y)]]\right)$

$= \exp[H(y) - H(y|x)]$

- The more realistic the image, the higher the inception score.

Frechet Inception distance

- distance b/w mean of a feature vector of pretrained & a new data point. & look at covariance over generated data & over training data.

$$d^2((m, C), (m_\omega, C_\omega)) \; ||m - m_\omega||_2^2 + \text{Tr}(C + C_\omega - 2(C C_\omega)^{1/2})$$

- In GANs, there isn't an optimal metric to optimize of fn.

Some thorey behind GANs

Bayes Optimal Discriminator

what the optimal discriminator given gen & true distri?

$$V(G,D) = E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p(z)}[\log(1-D(G(z)))]$$

$$= \int_x p_{data}(x)\log D(x)\,dx + \int_z p(z)\log(1-D(G(z)))$$

$$= \int_x p_{data}(x)\log D(x)\,dx + \int_x p_g(x)\log(1-D(x))\,dx$$

$$= \int_x [p_{data}(x)\log D(x) + p_g(x)\log(1-D(x))]\,dx$$

$\forall y\ [a\log y + b\log(1-y)]$, $\frac{dy}{dy} = 0 \Rightarrow y^* = \frac{a}{a+b}$ $\forall [a,b) \in \mathbb{R}^2 \setminus \{(0,0)\}$

$$\Rightarrow D^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$$ → optimal discri

$$\therefore V(G,D) = E_{x\sim p_{data}}[\log D^*(x)] + E_{x\sim p_g}[\log(1-D^*(x))]$$

$$= E_{x\sim p_{data}}\left[\log \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}\right] + E_{x\sim p_g}\left[\log \frac{p_g(x)}{p_{data}(x)+p_g(x)}\right]$$

$$= -\log(4) + KL\left(p_{data}\,\|\,\left(\frac{p_{data}+p_g}{2}\right)\right) + KL\left(p_g\,\|\,\left(\frac{p_{data}+p_g}{2}\right)\right)$$

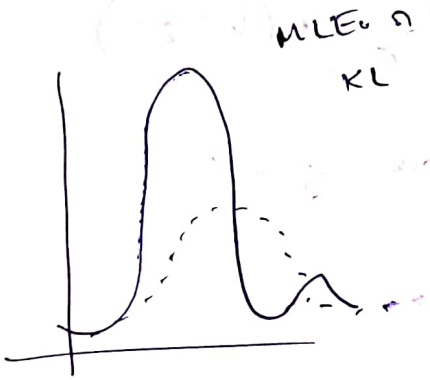
JS D. of p_{data} & p_g

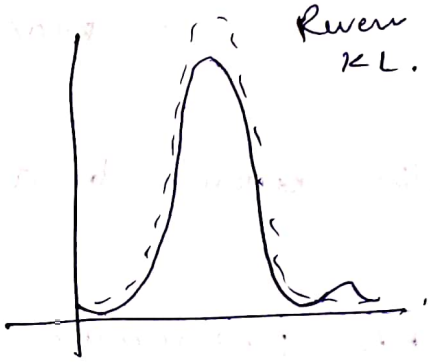
- Assuming the discri fully optimises, then the gen tries to optimise a type of KL div called JS D.

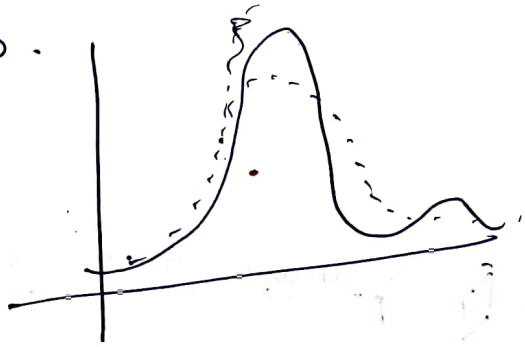
MLE tries to put fit — or find but scoring distri.

Reverse KL gives highest score on focusing on one mode.

JSD is something in b/w MLE & RKL.







Discriminator saturation

- If optimal D is reached, then the gen can't be trained as D generates down to 0 grad.

① So alternative optimisation is done.

② or Non saturating formulations.

$$L^{(n)} = -L^{D}$$

In this case

$$L^{(G)} = -L^{D} = \min_{G} E_{z \sim p(z)} \log\left(1 - D(G(z))\right)$$

$$L^{(D)} = -E_{x \sim p_{data}}[\log D(x)] - E_{z \sim p(z)}[\log(1 - D(G(z)))]$$

↓ non zero sum so won't get 0 grad.

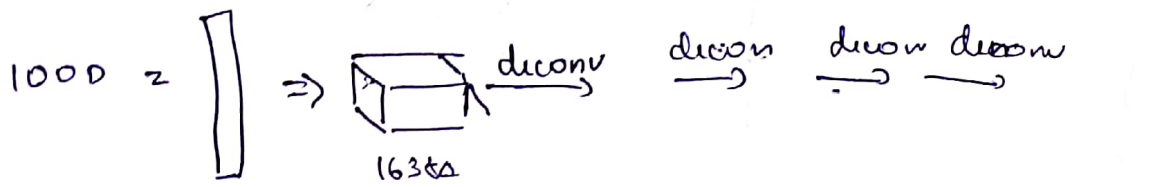
$$L^{(G)} = -E_{z \sim p(z)} \log(D(G(z))) =$$

$$= \max_{G} E_{z \sim p(z)} \log(D(G(z)))$$

This won't be a min max game.

GAN progression:

DCGAN



- spatially upsample
- channel downsample

- Most deconvs are normalized.

They are used to tie O/P $-1 \leq -1$.

$\left(\frac{x - }{127.5}\right)$

tanh at again O/P.

Because sampling time, have to go back to rgb

- for dis
  - remove max pooling, mean pooling
  - strided conv. & avg pooling at end.
  - leaky relu, relu for gen.
    (grads can become sparse)
- tanh gen, sig dis.
- BS used for prevent mode collapse
- BS not applied at o/p of G & i/p of D.
- small lr, small momen, bs 128

- vector arithmetic with DCGAN
- DC (discri) feature maps TL did better than supervised on CIFAR.

- Still unstable training
- Brittle architecture / hyperparam.

Improved training of GANs. (Salimas et al)

- feature matching
  - intermediate feature f in dis.
  - pass real data forward, collect f on dis, same on gen.

$$\| E_{x \sim p_{data}} f(x) - E_{z \sim p(z)} f(G(z)) \|^2$$

force feature of gen to match real data.

- minibatch discri

- Mode collapse → all noise in dishi is mapped to singh point by gen & discri is not aware this is happening as it looks at samples independently

- need to make discri aware that gen needs to produce diverse samples

- Let x_i be a sample in minibatch & it produces a feature $f(x_i)$

- A tensor is used to project $f(x_i)$ to a matrix

$$T \in R^{A \times B \times C} \qquad M_i \in R^{B \times C}$$

↓

B rows, C dims.

- Consider x_i & x_j,

$$c_b(x_i, x_j) \; 2\exp(L_1 \text{ norm}), \quad o(x_i)_b = \sum c_b(x_i, x_j)$$

- Do this for every single sample in minibatch. add all the distances. This gives a metric of how diff is a particular $f(x_i)$

- $o(x_i) = [o(x_i)_1 \ldots o(x_i)_B]$

$o(x) \in R^{n \times B}$

- to next layer of dis,

$$[f(x), o(x)]$$

Historical Averaging.

- inspired from Nash Equi.
- In a complicated 2 player game, with the issue of saddle point optim, it will basically be circling around in orbits.
- params in gen & disc kept close to historical avgs. Keep backups of params

One-sided label smoothing

- used in discr loss
- with optim D, $D(x) = \frac{(1-\alpha)p_{data}(x) + \beta p_{model}(x)}{p_{data}(x) + p_{model}(x)}$

this is removed because for mode collapsing, it'll help appearing, so it should not be included for gen.

Virtual Batch Norm

- use a ref bn to compute norm stats

$r = [r_1 \dots r_m]$ $\downarrow \mu, \sigma$

$x = [x^1 \dots x^m]$ $\frac{x-\mu}{\sigma}$

same for every subsq forward batch.

- But model could overfit if 1 mini batch is used
- To avoid this, append sample with ref bn $\rightarrow [x^i, r^1, \dots r^m]$

- Introduced semi-supervised for gans,
- idea is that if dis tells some img & predict, in addition, can predict category
- Then a classification in dis

- Inception score.

Earth Mover's distance

given a joint distribution P_r & P_g,

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)\sim\gamma}[\|x - y\|]$$

expectation over samples from joint distribution

dual version / both are intractable

If calculate for various joint distributions over all possible, then it is EMD

Kantorovich Rubinstein Duality:

$$W(P_r, P_g) = \sup_{\|f_L\|_1 \leq 1} E_{x\sim P_r}[f(x)] - E_{x\sim P_g} f(x)$$

supremum over all Leipzig 1 fns.

W-GAN

- ~~used~~ is EMD used in W-GAN
- how is it tractable?

given any family w

$$\max_{w \in W} \underbrace{E_{x \sim P_r}[f_w(x)] - E_{x \sim P_g}[f_w(x)]}$$

$$\leq \sup_{\|f_L\| \leq K} E_{x \sim P_r}[f(x)] - E_{x \sim P_g}[f(x)]$$

$$= K \cdot W(P_r, P_g)$$

↳ considered approx Leibnitz, lower bound is optimized on the EMD

- In the paper, they clip the weights to do this.

- $\sum_{1}^{m} f_w(x_i) - \frac{1}{m} \sum_{1}^{m} f_w(g_\theta(z^{(i)}))$

↓

this is clipped to make approx L-1.

- No clamping in this case, but a critic
- gen tries to maxi critic while discri tries to mini critic.

(c) WGAN-GP

- gradient penalty for Lip..
- read paper.
- (no weight clipping here)
- No batch norm in discri (followed in all gans now)

negatives

- training slower (due to ops)
- heuristic discri on grad penalty could be unstable.

Progressive GAN - Nvidia

- high quality samples
- progressively grow size of images for gen & dis, ex trained layers of lower levels

SN GAN (spectral normalization)

SN -> (duplicate)

Basic idea

weight $W = \frac{W}{\sigma(W)}$ -> largest singular value

do this for every weight matrix

- Hinge loss used, dis became something like a SVM clone

$$V_D(\hat{G}, D) = \mathop{E}_{x \sim q_{data}(x)} [\min(0, -1 + D(x))] +$$

$$\mathop{E}_{z \sim p(z)} [\min(0, -1 - D(\hat{G}(z)))]$$

$$V_G(G, \hat{D}) = -\mathop{E}_{z \sim p(z)} [\hat{D}(G(z))]$$

implementation

- power implementation for square matrix (Deepbays)

Self Attn GAN (SAGAN)

- Read paper.

BigGAN

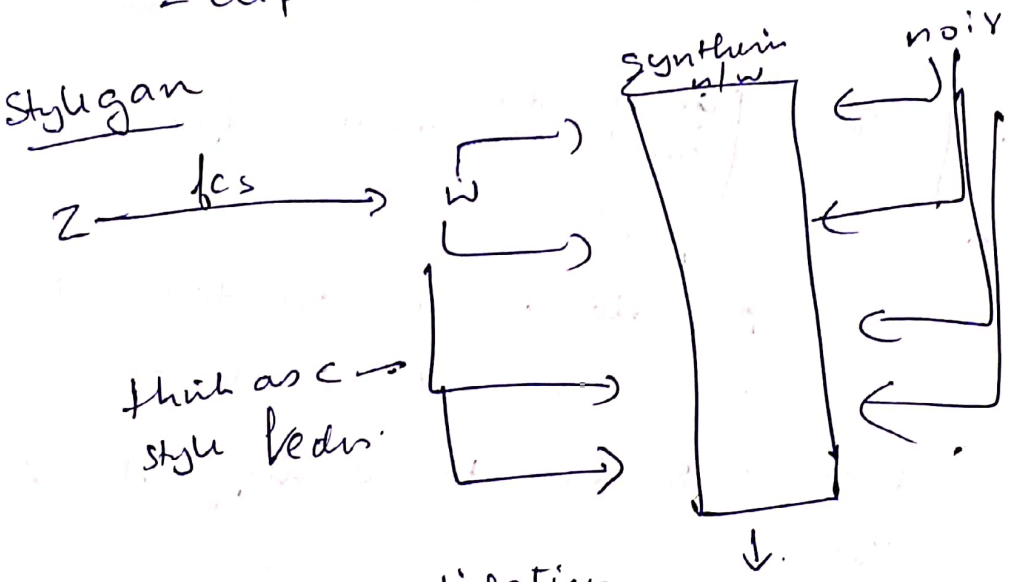
→ orthogonal reg

$$\beta || W^T W - I ||$$

(and don't penalize diagonal terms)

- truncation trick
  - clip sampling test time

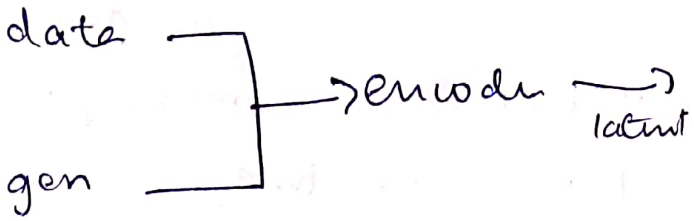
Stylegan



- Instance Normalisation

- BxHxWxC
- take avg stats over HxW

Information Bottleneck



here discri classifies in a lower dim

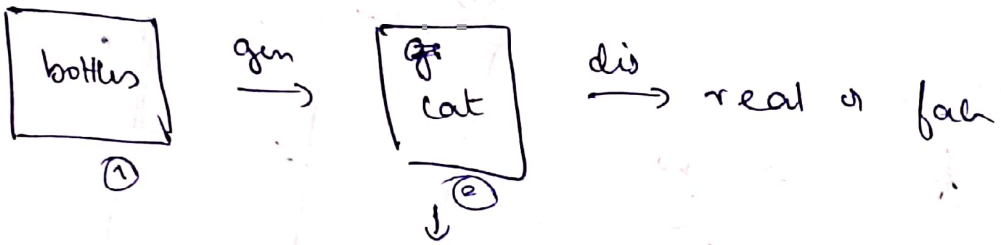
Conditional Gan | pix 2 pix

- modifying an image
  (black & white to colour)

black & white image → gen → gen image → dis → real / fake
                                              ↓
                                          valid colour image

gen synthesis fake colour images

limitation



gen did not do the task it was assigned. Bad objective.

- use both ① & ② to classify discri

Then loss

$$\arg\min_G \max_D E_{x,y}\left[\log D(x, G(x)) + \log\left(1 - D(x, y)\right)\right]$$

$$G^* = \arg\min_G \max_D L_{cGAN}(G, D) + \lambda L_1(G)$$

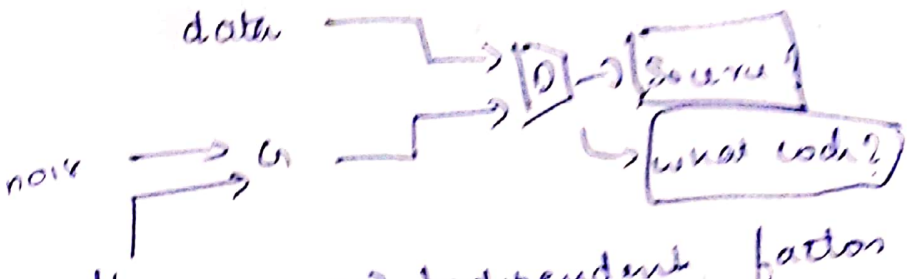
↓
reconstruction loss fo gen.

- shrinking discriminator - making it to focus on patches of image

GANs and augmentations

InfoGAN

- infer → type, rotation, width



→ Independent factors should maximally explain variations in generated image

$$\max_G I(c; x) = H(x) - H(x|c)$$
$$= H(c) - H(c|x)$$

$x = G(z, c)$

↓ variational lower bound to solve it as MI is intractable

$$I(c, G(z,c)) = H(c) - H(c|G(z,c))$$
$$= H(c) + E_{x \sim G(z,c)}\left[E_{c' \sim P(c|x)}[\log P(c'|x)]\right]$$
$$= H(c) + E_{x \sim G(z,c)}\left[E_{c' \sim P(c|x)}[\log Q(c'|x)] + D_{KL}(P(\cdot|x) \| Q(\cdot|x))\right]$$

$D_{KL} \geq 0$.

$$\geq H(c) + E_{x \sim G(z,c)}\left[E_{c' \sim P(c|x)}(\log Q(c'|x))\right]$$

↑ MLE

- In BigGAN,

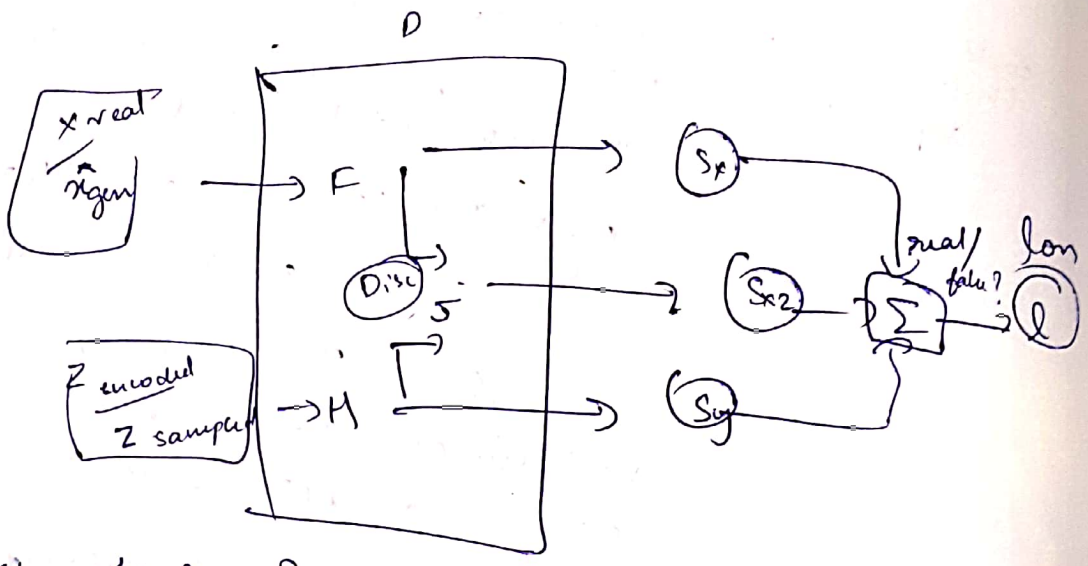
of the gen noise gz is changed

z → concat [[N(0,1)]^120 , Uniform(1024)];

↓

options for different classes

- BigGAN



Take find Encoder for Rep learning

- nearest neighbors on latent space give similar image

GANs as energy models

assign energy E(x) to every x

$$P(x) = \frac{1}{Z} e^{-E(x)} \qquad Z = \sum_{x \in X} e^{-E(x)}$$

Z → all possible world state (normalizing)

lower the energy, high the prob

- X can be really large

for an RGB 28×28×3 img,

* 256 ^ (28 * 28 * 3) = 256 ∞

- low energy for real looking image & high for those that don't look real.

$$\max_{\theta} E_{x \sim data}[\log P_\theta(x)]$$

$$= \max_{\theta} E_{x \sim P_{data}} \log \left(\frac{e^{-E_\theta(x)}}{Z(\theta)} \right)$$

$$= \max_{\theta} E_{x \sim P_{data}} [-E_\theta(x)] - \log Z(\theta)$$

(minimize energy) (hard to compute)

variational lower bound for Z

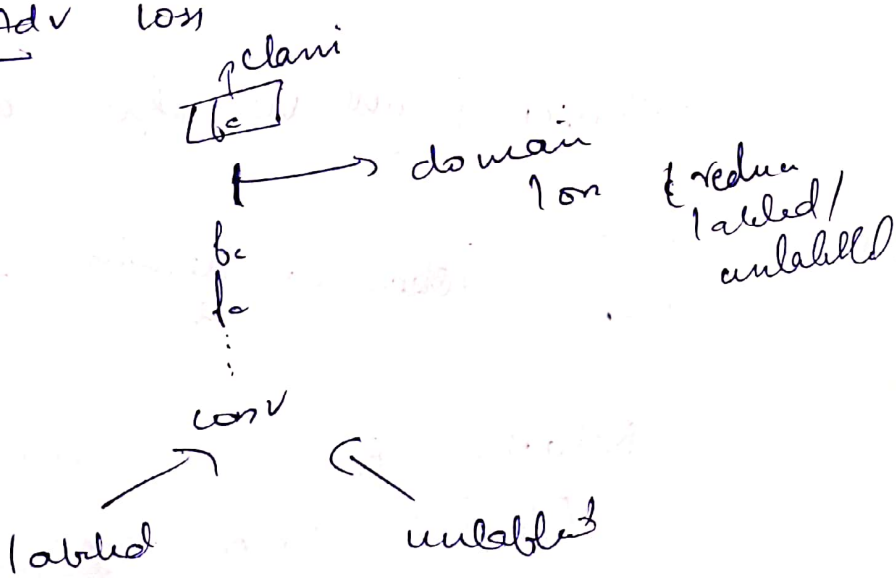
$$-\log Z \leq \min_{q_\phi} E_{x \sim q_\phi}[E_\theta(x)] - H(q_\phi)$$

$$= \max_{\theta} E_{x \sim P_{data}} [-E_\theta(x)] + \min_{q_\phi} E_{x \sim q_\phi} [E_\theta(x)] - H(q_\phi)$$

gan obj

- not easy to compute $H(q_\phi)$ but can be computed for - softmax

TL with Adv loss



- can also do this for farmers.

Self-Supervised

cognitive principles

- Reconstruct from corrupted
- Visual common sense tasks
- Constrastive learning

Reconstruct from corrupted

- denoising AE
  - Additive, isotropic gaussian noise
  - masking noise
  - Salt & pepper noise
  - stacked denoising AE (Literature on features)

- predicting missing piece
  - mask a piece, reconstruct
  - Joint loss

- predicting one view from another

 → color → denoising AE → grayscale

- Relative position of image patches
  - predict relative position

- jitter both spatially & colour wise
  - patches shouldn't overlap
    - to prevent chromatic aberration

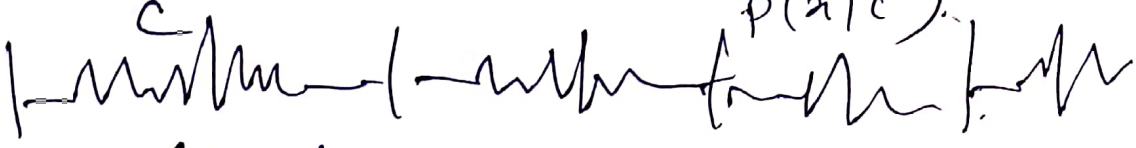
- Solving jigsaw puzzles
  - take patches from random crops, shuffle and predict proper ing.

- Rotation
  - 0, 180 works but on CIFAR

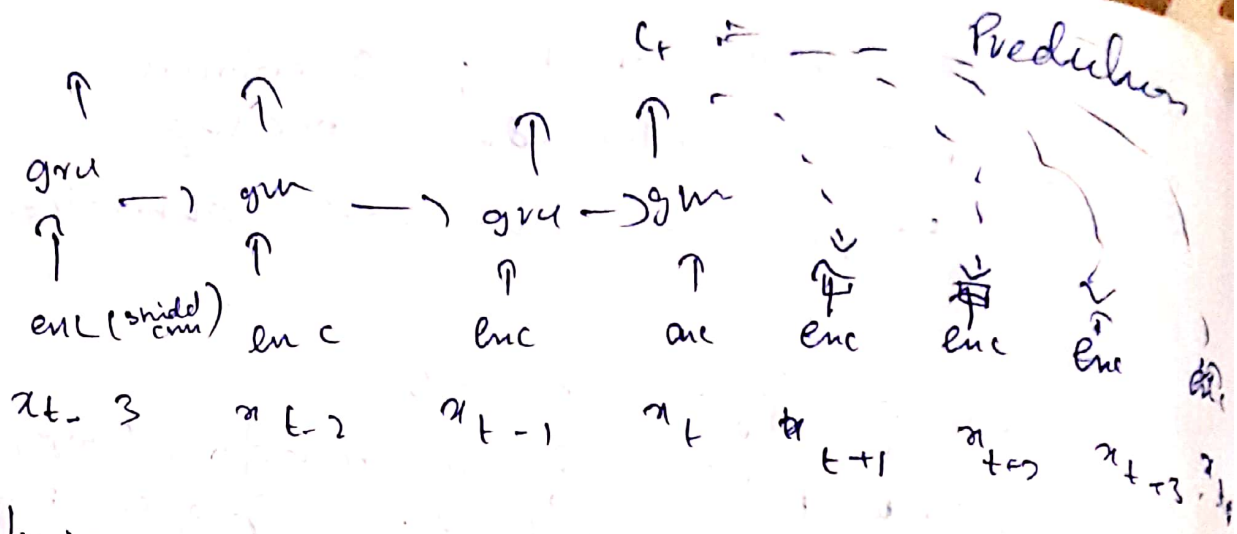
Contrastive learning

- word2vec
  - make count matrix
    - apply SVD
      - sparsity
      - computation
      - Infrequent word
      - noise from frequent
    - n-gram models

- Contrastive Predictive Coding



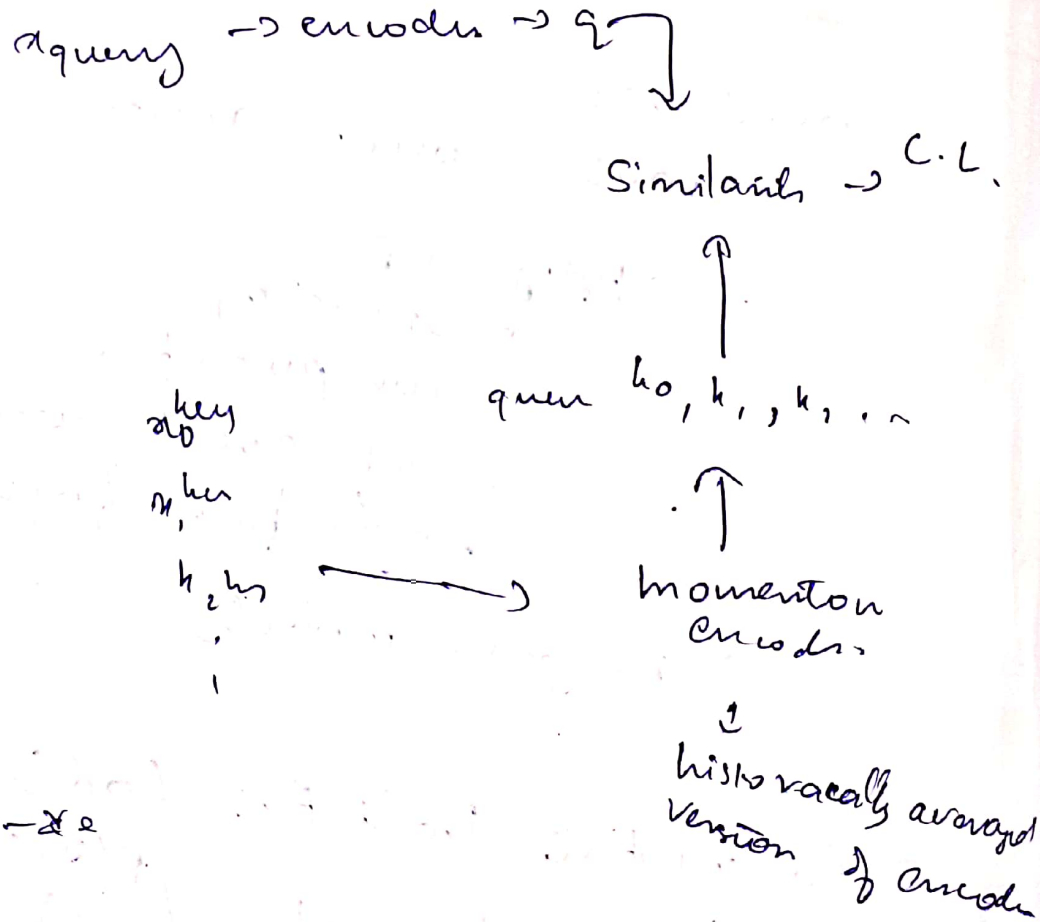
$$\max_{\theta} MI(c; z)$$



Instance Discrimination

- MOCO
- SimCLR (need full papers)

MOCO



Summary of unsupervised learning methods

Autoregressive
- NADE
- PixelRNN PixelCNN
- Wavenet
- Video Pixel Networks
- GPT 2

Success due to:
- large batch sizes, more compute, better stability in training
- wider, deeper
- conditional models on auxiliary variables
- Fewer assumptions
- Architecture
  - masked conv
  - dilated conv
  - transformers
- loss on Cross entropy

Future:
- model parallelism, larger models
- Same model for multiple modalities
- Fast sampling
- Architecture innovations

Negatives:
- no single layer of rep.
- sampling low for deployment
- No interpolations

Flow models

- NICE
- RLOW
- FlowGAN

LV models

- PixelVAE
- SketchRNN
- World models

Advantages

- Compressed Bottleneck rep
- Approx density estimation
- Disentangled reps

Dis:

- Blurry
- Factorised gaussian posterior or decoder assumption may limit

Future:

- Better decoders (cross entropy based, autoregressive)
- More powerful posteriors

GANs

- original
- DCGAN
- Wgan
- StyleGan
- Biggan

Future:
- more fine grained
- Video generation
- L-1 constraints (New approach)
- Arch (Upsampling, downs)
- Obj fun

Negatives:
- engineering tricks
- mode dropping
- Evaluation metrics

GANs & Density models?

~~mos~~ ~ same level of ~~indi~~ engineering details

- Blurry samples vs mode collapse (comprehension vs sample quality)
- GANs work well with less compute

For training density models:
- if only care abt density estimation, not Sampling, choose A.R.
- if both rep & sampling, use VAEs

When gans?
- good samples
- Large High quality images

Contrastive learning

- Dictionary look up task
- Predictive Coding & Instance Discri
  - end to end
  - momentum
  - CPCv2

MoCo
SimCLR

Future:

- gap not close b/w sdf-s & s
- down streak task ll ~~not~~ gains not that significant.

Modeling future in latent space

- internal world models that run on abstract space.

Semi Supervised

D_U: $x \sim p(x)$

↓

use this to make $p(x, y)$ better

D_S: $(x, y) \sim p(x, y)$

Core Concepts

- Confidence vs Entropy
  - Entropy minimisation
    - taking label data & making sure that classifier trainings on label data has minimal entropy on unlabeled data.

- Pseudo labeling
  - make classifier to predict labels on unlabeled data, take confidence predicts, correct them to extra ground truth. (a called sub training)

- Virtual Adversarial Training

- Label consistency
  - Make sure that augmentations of samples have the same class
  - Pi Model, Temporal Ensembling, Mean teach

- Regularisation
  - weight decay
  - Dropout
  - Data - Aug
  - Unsupervised Data Aug (UDA)
  - MixMatch

- Co-training / Self-training / Pseudo labelling. (Noisy student)

Entroply mini

pi Model