How to Train a GAN

Emily Denton, Martin Arjovsky, Michael Mathieu
New York University

Ian Goodfellow
Google

Soumith Chintala
Facebook AI Research

The stability of GANs

Timeline - the stability of GANs

Goodfellow et. al. “Generative Adversarial Networks”

2014

Timeline - the stability of GANs

Denton et. al. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”

2015

Timeline - the stability of GANs

model architecture generator

Denton et. al. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”

2015

Timeline - the stability of GANs

model architecture generator

visual inspection

Denton et. al. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”

2015

Timeline - the stability of GANs

model architecture generator

visual inspection

countless failed stability hacks

Denton et. al. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”

2015

Timeline - the stability of GANs

model architecture generator

visual inspection

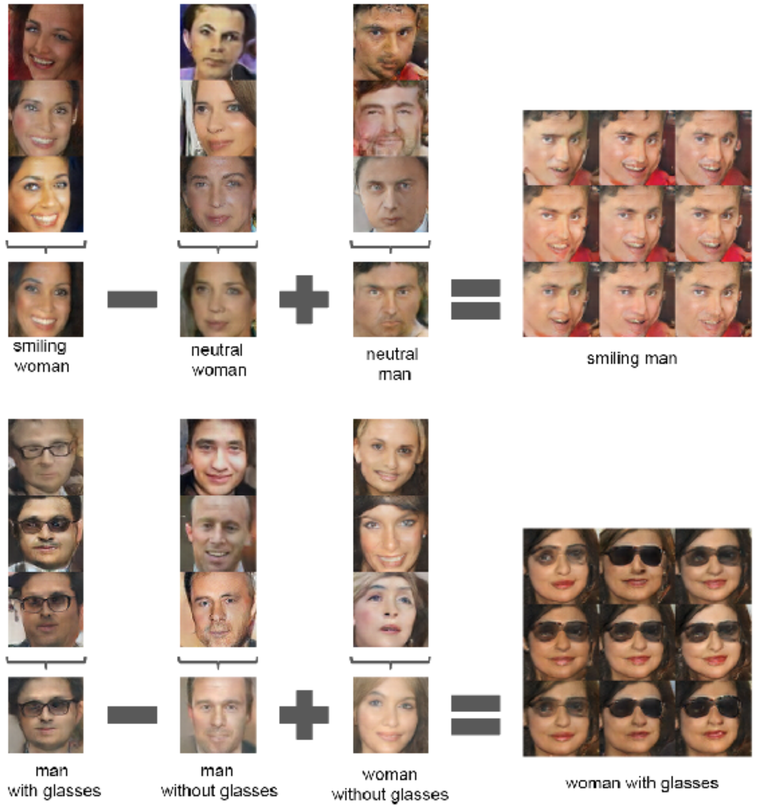
countless failed stability hacks

hope

Denton et. al. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”

2015

Timeline - the stability of GANs



countless hours finding stable models

stable upto 64x64

mode dropping

underfitting

Radford et. al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”

2015

Timeline - the stability of GANs

more heuristics

more stability

Salimans et. al. “Improved Techniques for Training GANs”

2015

Timeline - the stability of GANs

gradient norm regularization

Least-Squares
Boundary Equilibrium

Gulrajani et. al. "Improved Training of Wasserstein GANs"

Xudong et al. "Least squares generative adversarial networks."

Berthelot et. al. "Began: Boundary equilibrium generative adversarial networks."

2016-2017

Timeline - the stability of GANs

https://github.com/khanrc/tf.gans-comparison
by Junbum Cha

GANs comparison without cherry-picking

Implementations of some theoretical generative adversarial nets: DCGAN, EBGAN, LSGAN, WGAN, WGAN-GP, BEGAN, DRAGAN and CoulombGAN.

I implemented the structure of model equal to the structure in paper and compared it on the CelebA dataset and LSUN dataset without cherry-picking.

Comparison to Classification ConvNets

- Throw things at the wall and see what sticks
- Intuition is poorer
- Theoretical work is somewhat improving but still far away
- Objective validation metrics are not there yet

#1: Normalize the inputs

- normalize the images between -1 and 1
- Tanh as the last layer of the generator output
  - or some kind of bounds normalization

#2: Modified loss function (classic GAN)

- In papers people write min (log 1-D), but in practice folks practically use max log D
  - because the first formulation has vanishing gradients early on
  - Goodfellow et. al (2014)
- In practice:
  - Flip labels when training generator: real = fake, fake = real

#2: Modified loss function (classic GAN)

- In papers people write min (log 1-D), but in practice folks practically use max log D
  - because the first formulation has vanishing gradients early on
  - Goodfellow et. al (2014)
- In practice:
  - Flip labels when training generator: real = fake, fake = real

LOT OF NEW LOSS FORMULATIONS

#2: Modified loss function (classic GAN)

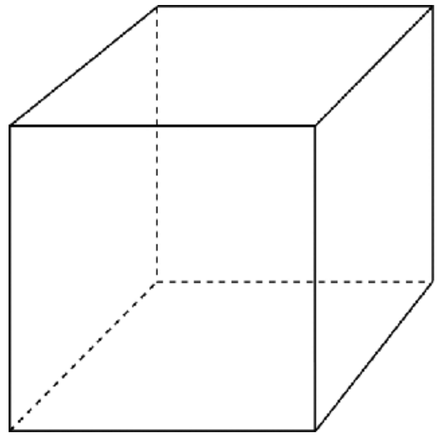
https://github.com/hwalsuklee/tensorflow-generative-model-collections

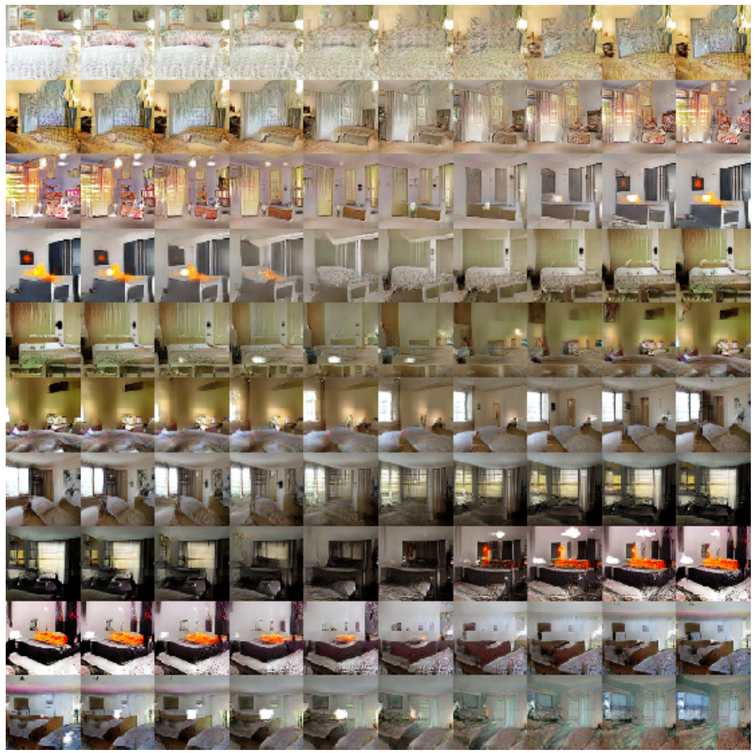
https://github.com/znxlwm/pytorch-generative-model-collections

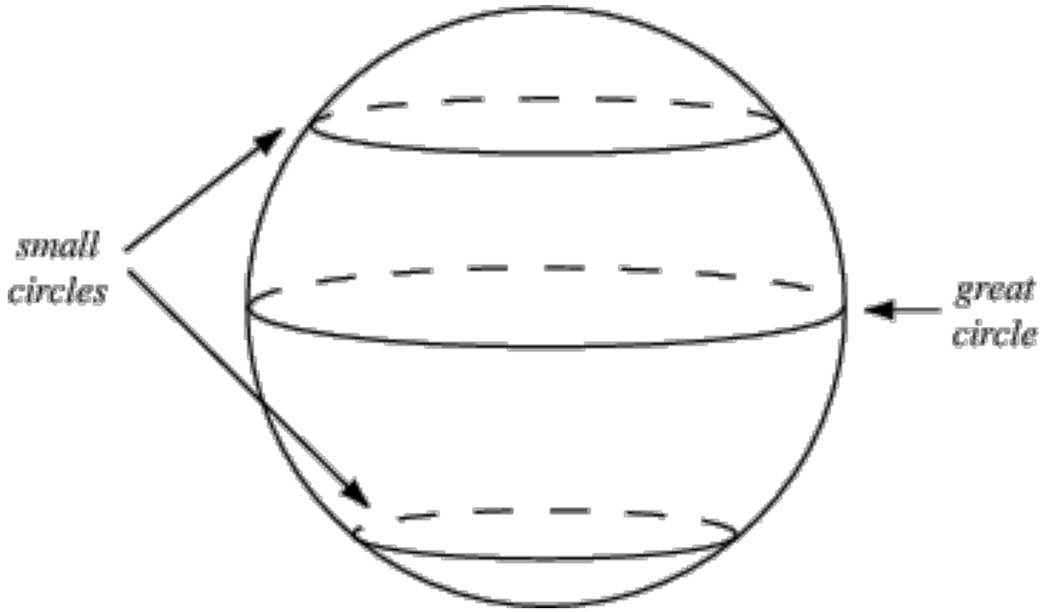
Name	Paper Link	Value Function
GAN	Arxiv	$L_D^{GAN} = E\left[\log\left(D(x)\right)\right] + E\left[\log\left(1 - D(G(z))\right)\right]$ $L_G^{GAN} = E\left[\log\left(D(G(z))\right)\right]$
LSGAN	Arxiv	$L_D^{LSGAN} = E[(D(x) - 1)^2] + E[D(G(z))^2]$ $L_G^{LSGAN} = E[(D(G(z)) - 1)^2]$
WGAN	Arxiv	$L_D^{WGAN} = E[D(x)] - E[D(G(z))]$ $L_G^{WGAN} = E[D(G(z))]$ $W_D \leftarrow clip\_by\_value(W_D, -0.01, 0.01)$
WGAN-GP	Arxiv	$L_D^{WGAN\_GP} = L_D^{WGAN} + \lambda E[(\lvert\nabla D(\alpha x - (1 - \alpha G(z)))\rvert - 1)^2]$ $L_G^{WGAN\_GP} = L_G^{WGAN}$
DRAGAN	Arxiv	$L_D^{DRAGAN} = L_D^{GAN} + \lambda E[\left(\lvert\nabla D(\alpha x - (1 - \alpha x_p))\rvert - 1\right)^2]$ $L_G^{DRAGAN} = L_G^{GAN}$
CGAN	Arxiv	$L_D^{CGAN} = E\left[\log\left(D(x, c)\right)\right] + E\left[\log\left(1 - D(G(z), c)\right)\right]$ $L_G^{CGAN} = E\left[\log\left(D(G(z), c)\right)\right]$
infoGAN	Arxiv	$L_{D,Q}^{infoGAN} = L_D^{GAN} - \lambda L_I(c, c')$ $L_G^{infoGAN} = L_G^{GAN} - \lambda L_I(c, c')$
ACGAN	Arxiv	$L_{D,Q}^{ACGAN} = L_D^{GAN} + E[P(class = c\lvert x)] + E[P(class = c\lvert G(z))]$ $L_G^{ACGAN} = L_G^{GAN} + E[P(class = c\lvert G(z))]$
EBGAN	Arxiv	$L_D^{EBGAN} = D_{AE}(x) + \max(0, m - D_{AE}(G(z)))$ $L_G^{EBGAN} = D_{AE}(G(z)) + \lambda \cdot PT$
BEGAN	Arxiv	$L_D^{BEGAN} = D_{AE}(x) - k_t D_{AE}(G(z))$ $L_G^{BEGAN} = D_{AE}(G(z))$ $k_{t+1} = k_t + \lambda(\gamma D_{AE}(x) - D_{AE}(G(z)))$

#3: Use spherical z

- interpolation via great circle
- Tom White “Sampling Generative Networks”
  - https://arxiv.org/abs/1609.04468

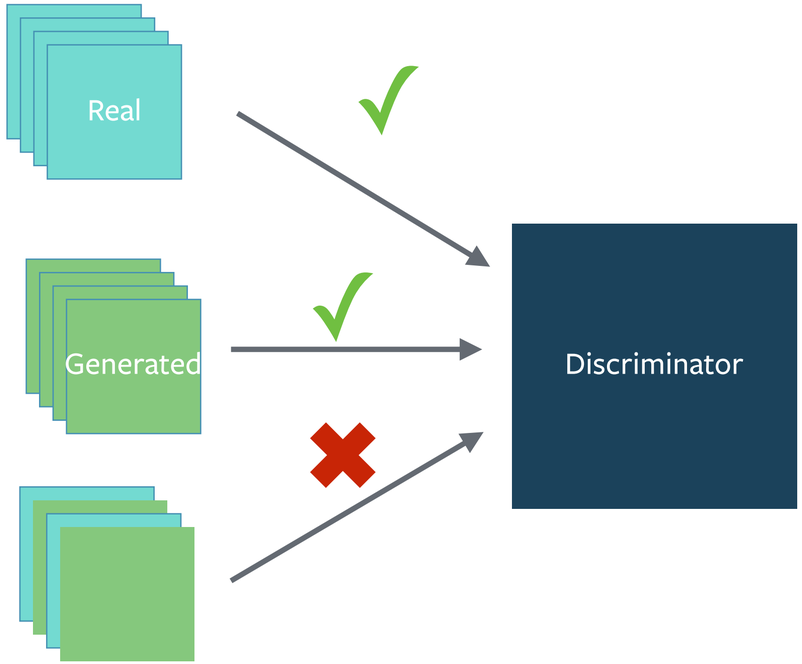






#4: BatchNorm

- different batches for real and fake
- when batchnorm is not an option use instance norm



#5: Avoid Sparse Gradients: ReLU, MaxPool

- the stability of the GAN game suffers
- LeakyReLU (both G and D)
- Downsampling: Average Pooling, Conv2d + stride
- Upsampling: PixelShuffle, ConvTranspose2d + stride
  - PixelShuffle: https://arxiv.org/abs/1609.05158

#6: Soft and Noisy Labels

- Label Smoothing
- making the labels the noisy a bit for the discriminator, sometimes

  -Salimans et. al. 2016

#7: Architectures: DCGANs / Hybrids

- DCGAN when you can

- if you cant use DCGANs and no model is stable,
- use a hybrid model : KL + GAN or VAE + GAN

- ResNets from WGAN-gp also work pretty well (but are very slow)
  - https://github.com/igul222/improved_wgan_training

- Width matters more than Depth

#8: Stability tricks from RL

- Experience replay
- Things that work for deep deterministic policy gradients
- See Pfau & Vinyals (2016)

#9: Optimizer: ADAM

- optim.Adam rules!
  - See Radford et. al. 2015
- [MMathieu] Use SGD for discriminator and ADAM for generator

#10: Use Gradient Penalty

- Regularize the norm of the gradients
  - multiple theories on why this is useful (WGAN-GP, DRAGAN, Stabilizing GANs by Regularization etc.)

#11: Dont balance via loss statistics (classic GAN)

- Dont try to find a (number of G / number of D) schedule to uncollapse training

- while lossD > X:
  - train D
- while lossG > X:
  - train G

#12: If you have labels, use them

- if you have labels available, training the discriminator to also classify the samples: auxillary GANs

#13: Add noise to inputs, decay over time

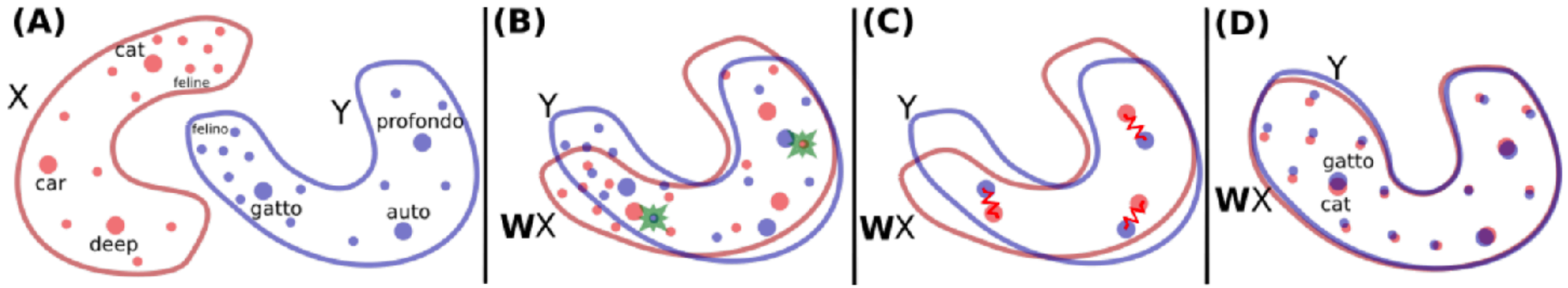
- Add some artificial noise to inputs to D (Arjovsky et. al., Huszar, 2016)
  - http://www.inference.vc/instance-noise-a-trick-for-stabilising-gan-training/
  - https://openreview.net/forum?id=Hk4_qw5xe
- adding gaussian noise to every layer of generator (Zhao et. al. EBGAN)
- Improved GANs: OpenAI code also has it (commented out)

#14: Train discriminator more

- especially when you have noise
- hard to find a schedule of number of D iterations vs G iterations
- WGAN/WGAN-gp papers suggest 5x D iterations per G iteration

#15: Avoid discrete spaces

- Pose the generation as a continuous prediction



Conneau, Lample et. al. "Word Translation Without Parallel Data"

Conditional GANs

#16: Discrete Variables

- Use an Embedding layer
- Add as additional channels to images
- Keep embedding dimensionality low and upsample to match image channel size

Conclusion

- Model stability is improving
- Theory is improving
- Hacks are a stop-gap