How to Train a GAN

Soumith Chintala
Facebook AI Research

Emily Denton, Martin Arjovsky, Michael Mathieu
New York University

Timeline - 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

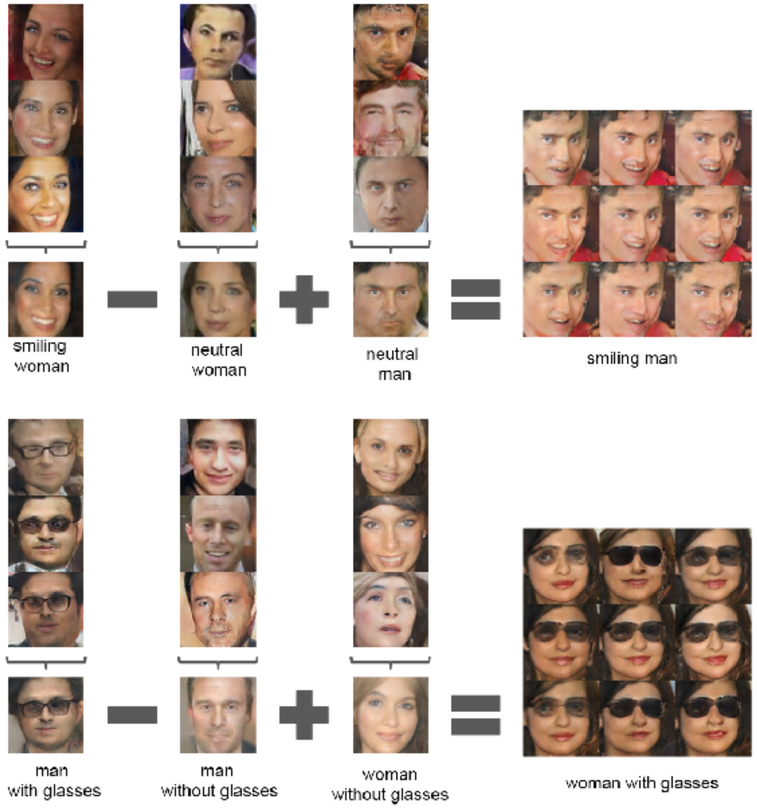
countless failed stability hacks

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

Comparison to Classification ConvNets

- Throw things at the wall and see what sticks
- Intuition is poorer
- No objective validation

#1: Normalize the inputs

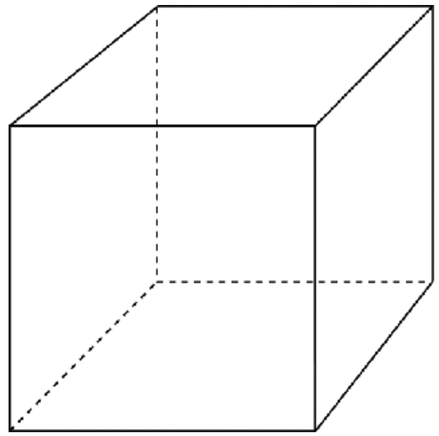
- normalize the images between -1 and 1
- Tanh as the last layer of the generator output

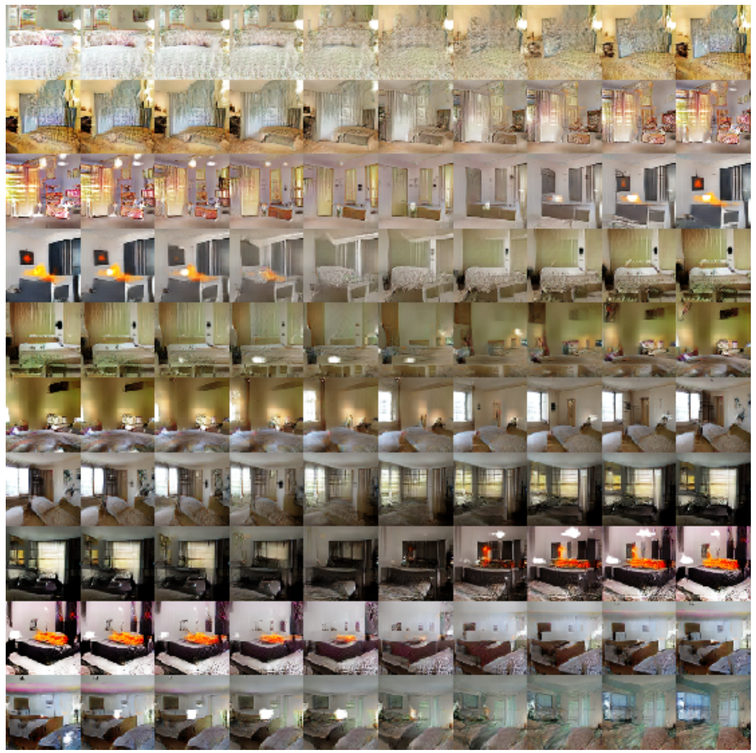
#2: Modified loss function

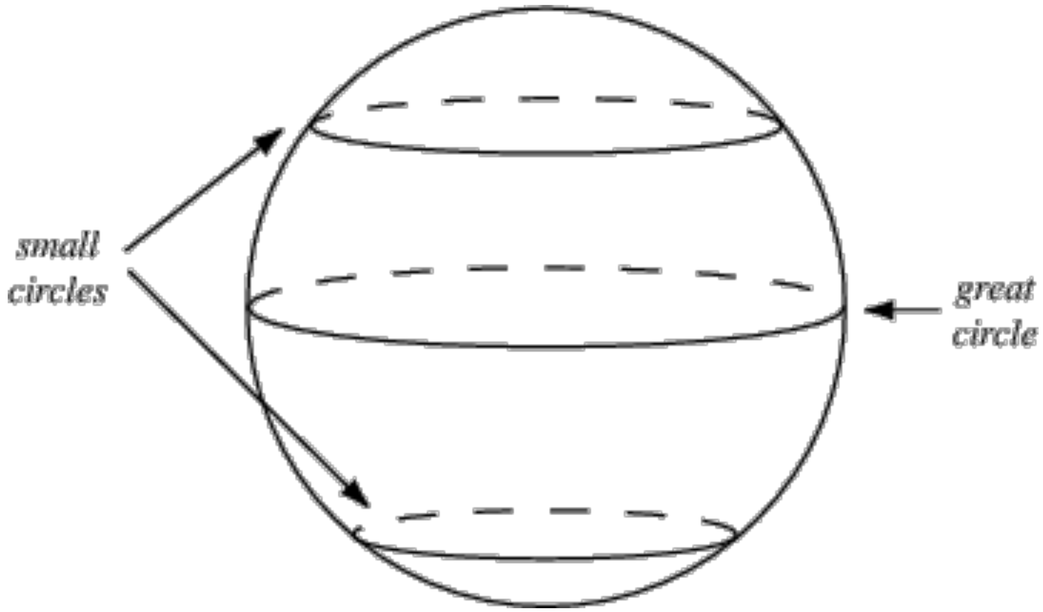
- In papers people write min (log 1-D), but in practice folks practically use max log D
  - because the first formulation has vanishing gradients early on
  - Goodfellow et. al (2014)
- In practice:
  - Flip labels when training generator: real = fake, fake = real

#3: Use spherical z

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

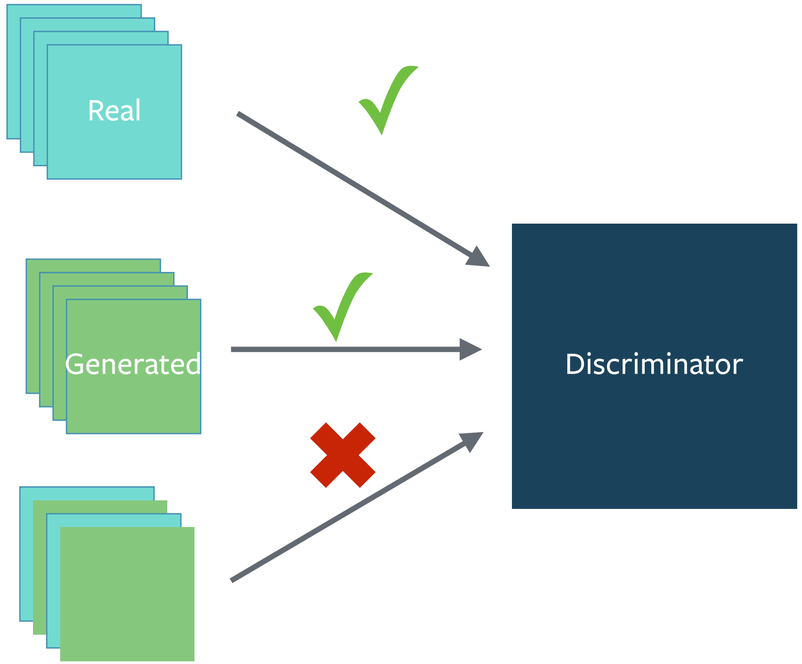






#4: BatchNorm

- 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: DCGANs / Hybrid models

- DCGAN when you can

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

#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: Track failures early

- D loss = 0
- check norms of gradients: if they are over 100, things are screwing up
- when things are working, D loss has low variance and goes down over time vs having huge variance and spiking
- if loss of generator steadily decreases, then it's fooling D with garbage

#11: Dont balance via loss statistics

- 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 (sometimes)

- especially when you have noise
- hard to find a schedule of number of D iterations vs G iterations

#15: Batch Discrimination

- Mixed results

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