Recent Advances in GANs

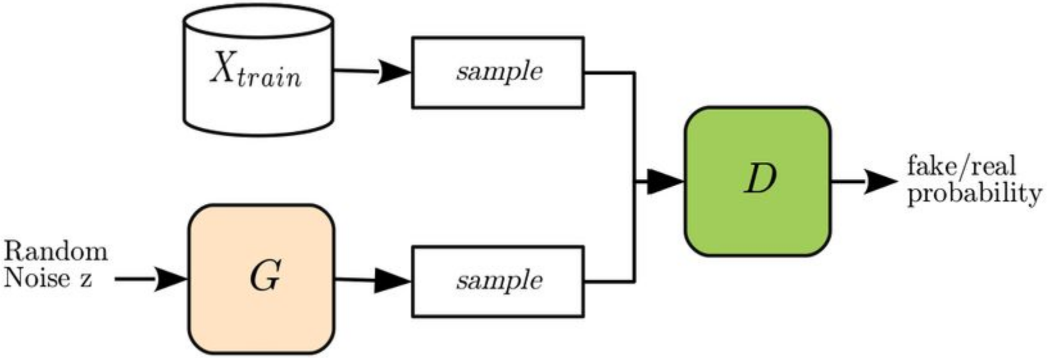
Quinn Frank, Cole Juracek

Background: GANs

- Proposed by Goodfellow, et al. (2014)
- Consists of a generator *G* which maps latent vector ***z*** $\longrightarrow$ ***image*** and a discriminator *D* which maps ***image*** $\longrightarrow$ [0, 1] (probability that image is real)
- **G** and **D** train each other via a minimax game

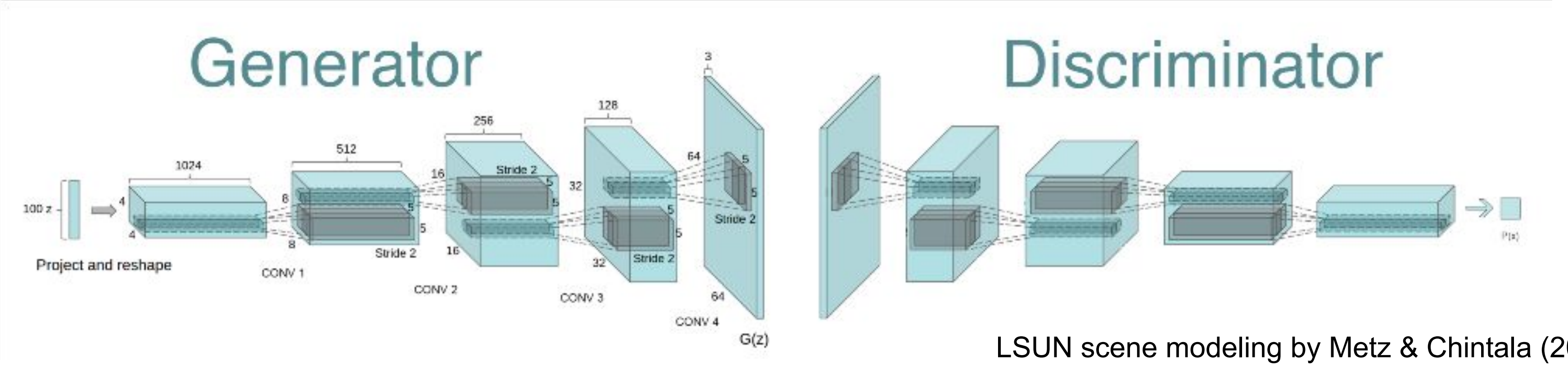
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

- Optimizes the Jensen-Shannon divergence between distributions p_{data} and $p_{\boldsymbol{z}}$



DCGANs

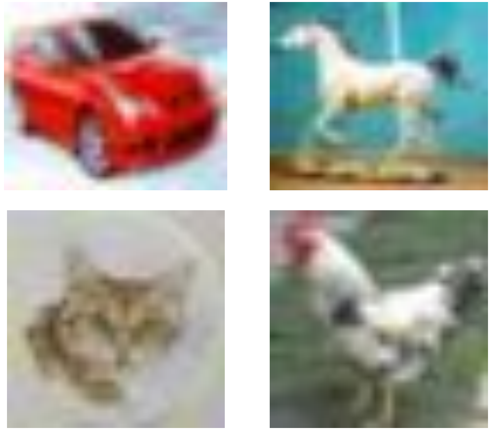
- Facilitate more stable GAN training with large, multi-channel images
- Set of architectural heuristics:
  - Use only convolutional layers in both networks, with no pooling operations
  - Batch normalization everywhere except output of *G* and input of *D* (**prevents mode collapse**)
  - ReLU + Tanh in generator; Leaky ReLU + Sigmoid in discriminator



LSUN scene modeling by Metz & Chintala (2016)

CIFAR-10 Architecture

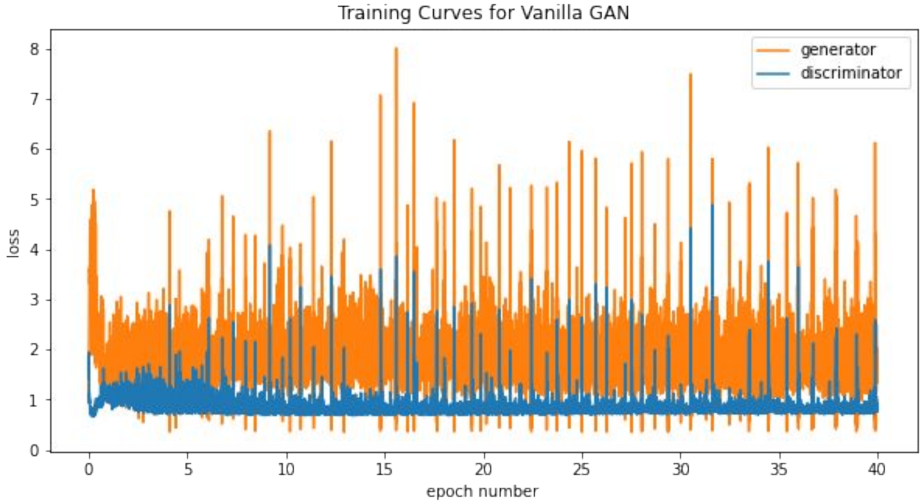
- Benchmark dataset of 32 x 32 images classified into 10 categories
- We use a version of the original DCGAN architecture modified for images of this size



type	kernel/stride/pad	input size
conv tr.	4 x 1 x 0	100 x 1 x 1
conv tr.	4 x 2 x 1	512 x 4 x 4
conv tr.	4 x 2 x 1	256 x 8 x 8
conv tr.	4 x 2 x 1	128 x 16 x 16
tanh	activation	3 x 32 x 32

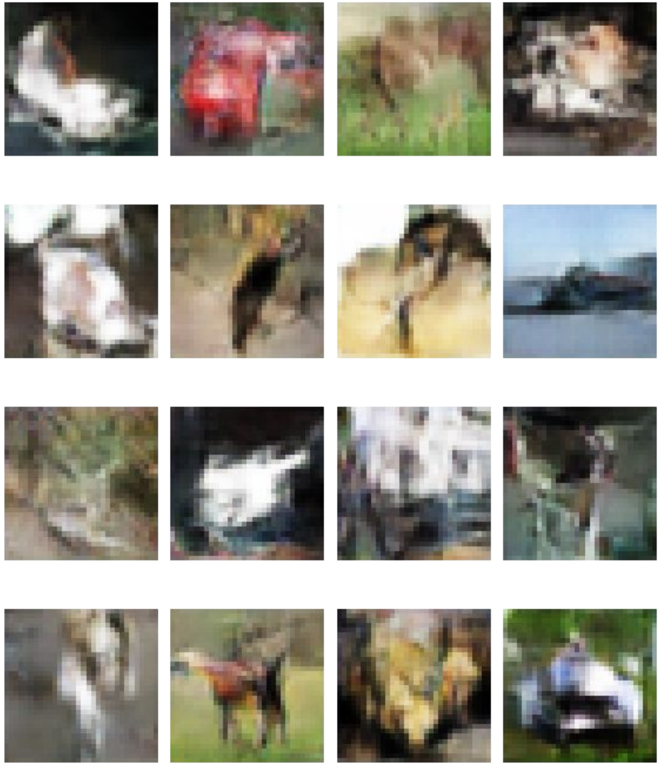
type	kernel/stride/pad	input size
conv	4 x 2 x 1	3 x 32 x 32
conv	4 x 2 x 1	64 x 16 x 16
conv	4 x 2 x 1	128 x 8 x 8
conv	4 x 1 x 0	256 x 4 x 4
sigmoid	activation	1 x 1 x 1

Vanilla GAN Training



Linear interpolation in the latent space (boat to horse transformation)

Sampled images from trained DC generator:



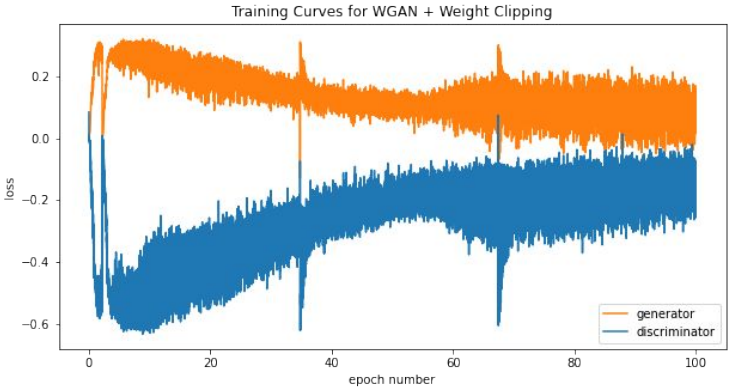
Wasserstein GANs

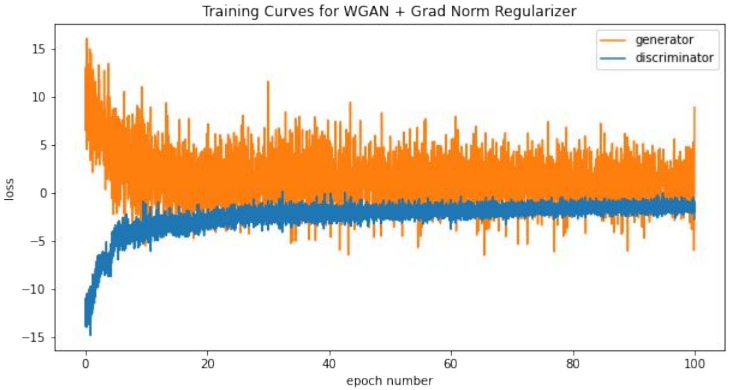
- Optimizes the Earth Mover distance (Wasserstein loss) between p_{data} and $p_{\mathbf{z}}$
  - Intractable, so maximize an approximation with respect to all 1-Lipschitz discriminators *f*

$$\max_{w \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim p(z)}[f_w(g_\theta(z))]$$

- To enforce Lipschitz continuity, use **weight clipping** (Arjovsky, et al. 2017)
  - Simply restrict the weights of *D* to some box [-0.01, 0.01] after each update
- Alternatively, can use **gradient norm penalization** (Gulrajani, et al. 2017)
  - Interpolate some point **x’** between real and simulated image data, then regularize via the L2-norm of the gradient of the discriminator with respect to **x’**
- Requires a linear activation at the output of the discriminator

WGAN Training



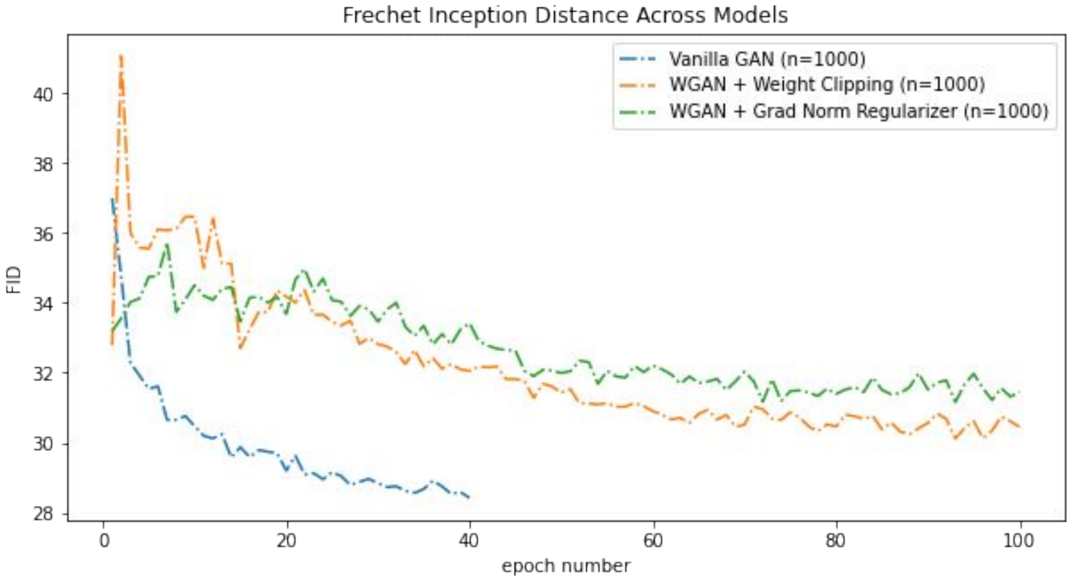


Sampled images from weight clipping *(top)* and gradient norm penalty *(bottom)*. The training curves of both WGAN models are more stable than vanilla, but the generated images are not noticeably better after many epochs.

Evaluating Generated Images

- Implemented the Frechet Inception Distance (FID) to measure image quality
  - Embeds images to 2048-dim feature space through a pre-trained Inception v3 network
  - Embed samples from p_{data} and $p_{\mathbf{z}}$, calculate Frechet distance between distributions
  - Requires large number of samples to reduce variance in estimate

- FID estimates were taken after each epoch
- Vanilla DCGAN achieved higher quality images faster than either regularized model; though difficult to compare timescales



Other Regularization Techniques

- Can combine WGAN with **spectral normalization,** which enforces a Lipschitz constraint by normalizing the *spectral norm* of the generator weights
- **Stable rank normalization** is an improved version of the above that both normalizes both the *spectral norm* and the *stable rank*
  - Also applies more generally to non-GAN networks
- Different architectures, like the Conditional GAN, have been shown to stabilize training, as well

- Limitations/improvements?