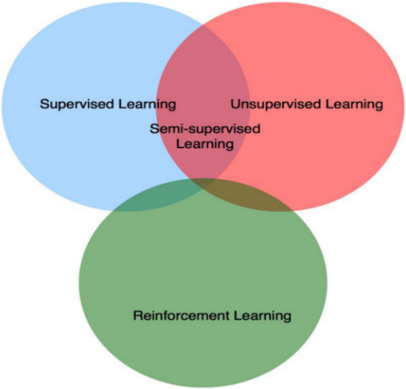
Generative Adversarial Networks (GANs)

The coolest idea in Machine Learning in the last twenty years - Yann Lecun

Overview

- **Generative Adversarial Networks (GANs)**
- 3D GANs
- Domain Adaptation

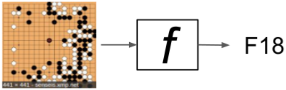
Introduction



From David silver, Reinforcement learning (UCL course on RL, 2015).

Supervised Learning

- Find deterministic function f: y = f(x), x:data, y:label



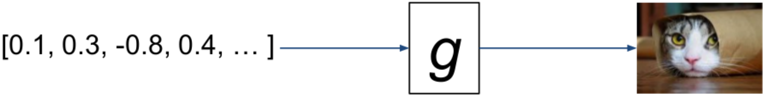
Unsupervised Learning

"Most of human and animal learning is unsupervised learning. If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake. We know how to make the icing and the cherry, but we do not know how to make the cake. We need to solve the unsupervised learning problem before we can even think of getting to true AI." - Yann Lecun

"You cannot predict what you cannot understand" - Anonymous

Unsupervised Learning

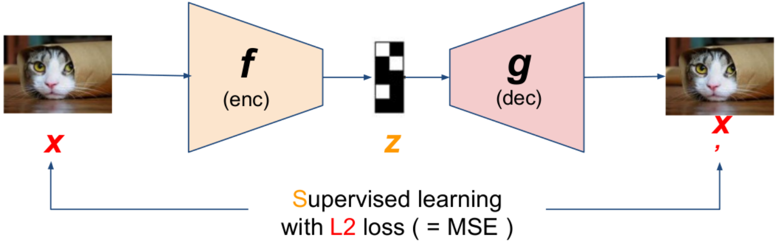
- More challenging than supervised learning. No label or curriculum.
- Some NN solutions:
  - Boltzmann machine
  - AutoEncoder
  - Generative Adversarial Networks

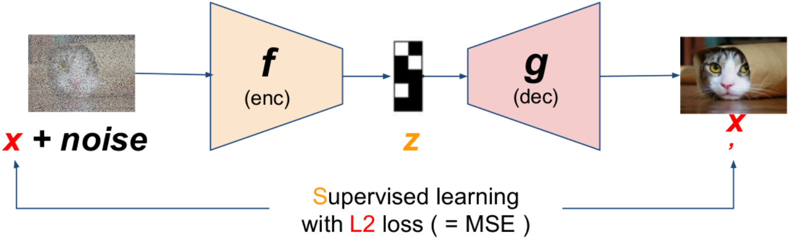


Unsupervised Learning vs Generative Model

- $z = f(x)$ vs. $x = g(z)$
- $P(z|x)$ vs $P(x|z)$

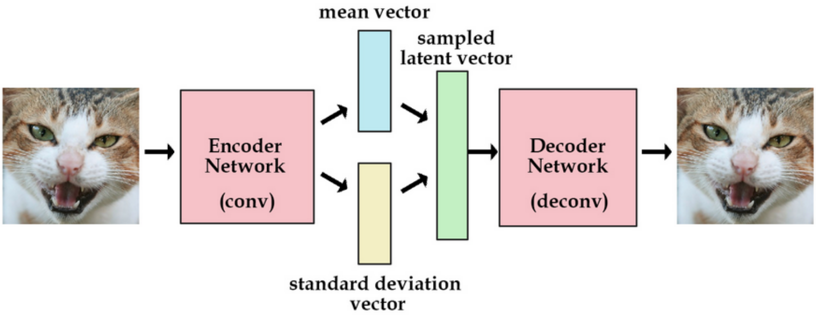
- Use data itself as label





x + noise
f
(enc)
z
g
(dec)
x
'
Supervised learning
with L2 loss (= MSE)

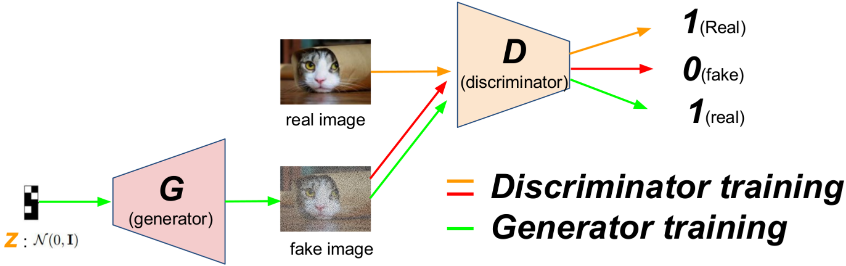
Variational Autoencoder



Generative Adversarial Networks

- Ian Goodfellow et al, "Generative Adversarial Networks", 2014.
- Mini-Max game based on Nash Equilibrium
- Hard to train. No guaranteed equilibrium

Generative Adversarial Networks



Generative Adversarial Networks

Value of   Expectation   prob. of D(real)   prob. of D(fake)

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

Minimize G   Maximize D   x is sampled from real data   z is sampled from N(0, I)   fake

Generative Adversarial Networks

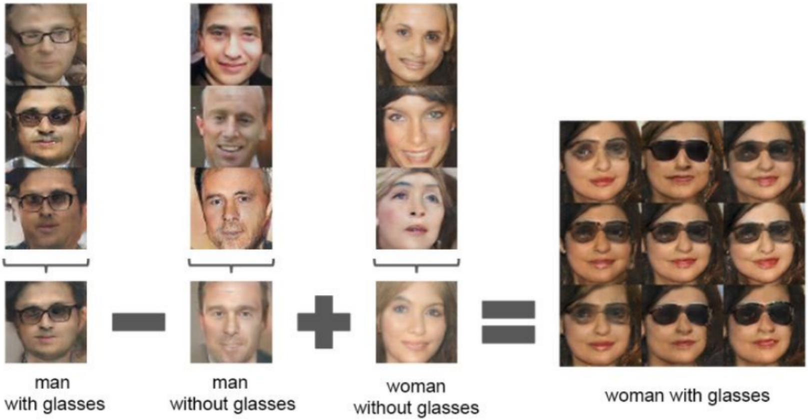
Result

- DCGAN used the following tricks:
    - Use LeakyRelu instead of RELU
    - Use Batchnorm in both generator and discriminator
    - Adam optimizer (lr = 0.0002, beta1 = 0.5)

Generative Adversarial Networks

Results

Latent Arithmetic



- GAN hacks proposed by Soumit Chintala et al
- Sample from Gaussian instead of uniform.
- Use batchnorm in both generator and discriminator
- Stability tricks from RL
- Dropouts in both train and test time in both G and D

- Image Generation,Progressive GAN (NVIDIA)

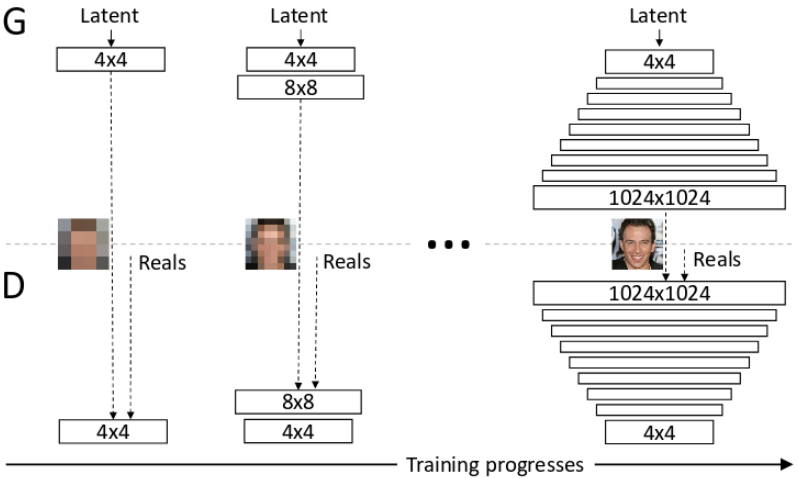


Figure: Results by Progressive GAN

- Translating Image, Perarnau et al, Invertible conditional GANsfor image editing.

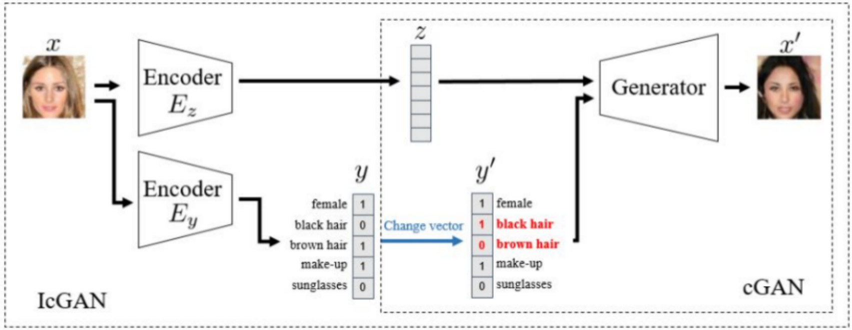


Figure: Results of ICGAN

- Stack GAN, Zhang et al, Text to Photo Realisitc Image Synthesis.

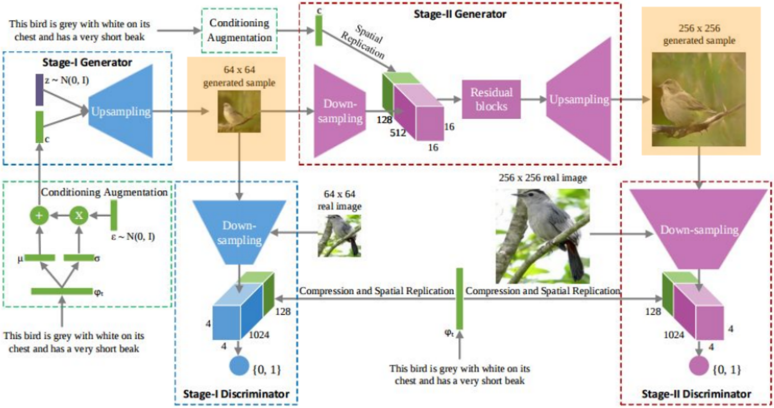


Figure: Stack GAN archtecture

Generative Adversarial Networks
Applications

Figure: Stack GAN Results

Overview

- Generative Adversarial Networks (GANs)
- **3D GANs**
- Domain Adaptation

3D GANs

- How to extend the GANs for 3D shapes?
- Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling (MIT)

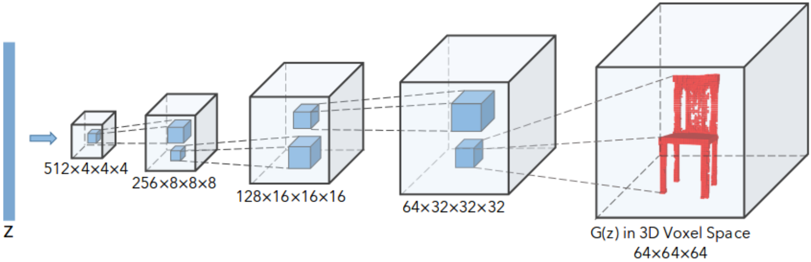


Figure: The architecture of generator in 3D GAN

3D GANs

Results

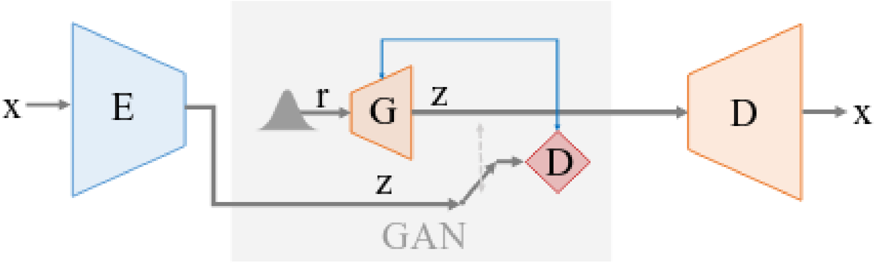
- Provided smooth interpolations
- Video: `https://youtu.be/mfx7uAkUtCI`
- Disrciminator can be used for classification (with minimal supervision)

Supervision	**Pretraining**	**Method**	**Classification (Accuracy)**	
			ModelNet40	ModelNet10
Category labels	ImageNet	MVCNN [Su et al., 2015a]	90.1%	-
		MVCNN-MultiRes [Qi et al., 2016]	**91.4%**	-
	None	3D ShapeNets [Wu et al., 2015]	77.3%	83.5%
		DeepPano [Shi et al., 2015]	77.6%	85.5%
		VoxNet [Maturana and Scherer, 2015]	83.0%	92.0%
		ORION [Sedaghat et al., 2016]	-	**93.8%**
Unsupervised	-	SPH [Kazhdan et al., 2003]	68.2%	79.8%
		LFD [Chen et al., 2003]	75.5%	79.9%
		T-L Network [Girdhar et al., 2016]	74.4%	-
		VConv-DAE [Sharma et al., 2016]	75.5%	80.5%
		3D-GAN (ours)	**83.3%**	**91.0%**

Figure: Classification Results

GAN for point clouds

- Learning Representations and Generative Models for 3D Point Clouds, Panos et al 2017.
- Generator and Decoder consists of FC layers. The Autoencoder has 1-D convolutions.
- Used a pretrained autoencoder based on EMD (or Chamfer) loss to encode the images first.



GAN for point clouds

Results

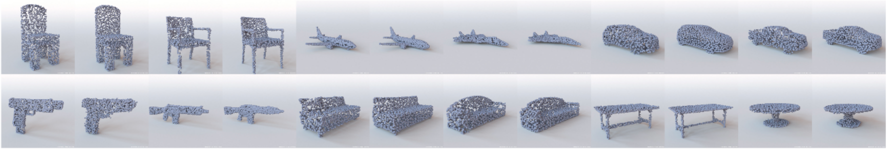


Figure: Results on test dataset. Left shows ground truth and the right image shows the reconstruction

GAN for point clouds

Results

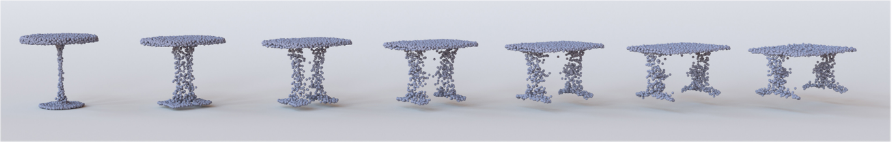


Figure: Interpolation

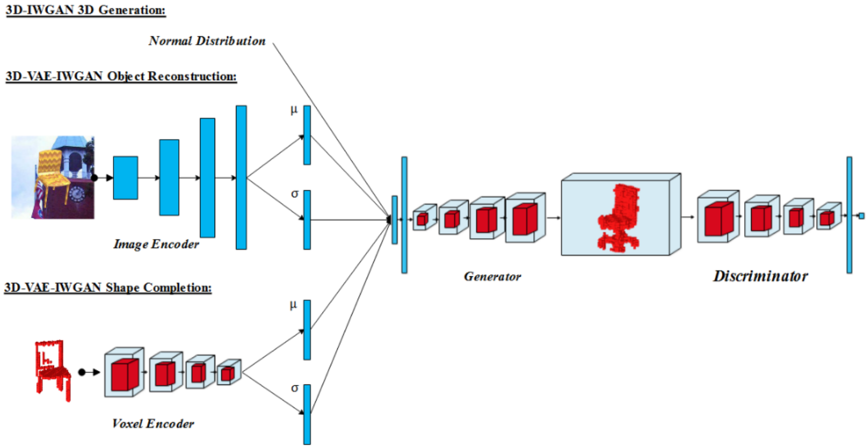


Figure: Improved Wasserstein GAN

3D Object Generation and Reconstruction

Results

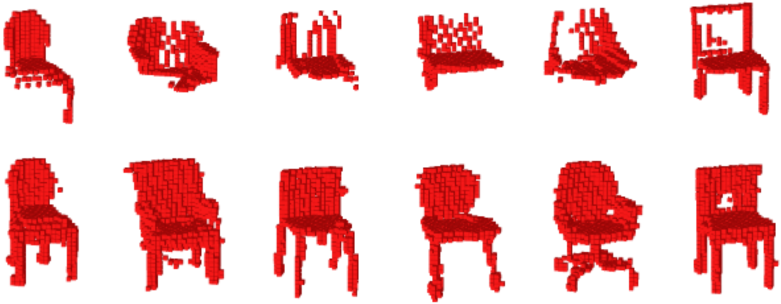


Figure: 3D Shape Completion

Overview

- Generative Adversarial Networks (GANs)
- 3D GANs
- **Domain Adaptation**

Domain Adaptation

Motivation

- Large annotated data is very expensive to obtain. (ImageNet, MS COCO)
- Alternative? Use synthetic data. (do not generalize to real images)
- Domain Adaptation: Transfer knowledge from source domain (labelled) to target domain (no labels).

- Unsupervised Domain Level Adaptation with GANs, K. Bousmalllis et al, 2017
- Goal is to come up with a classifier trained on source domain and can generalize to target domain.
- Previous works coupled the classifier and the task.

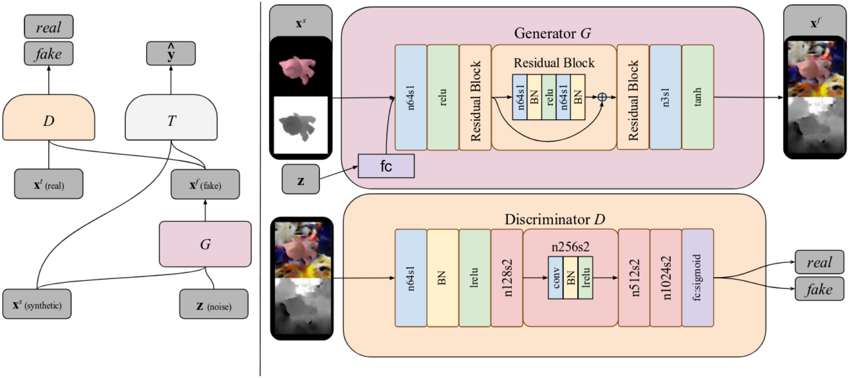


Figure 2. An overview of the model architecture. On the left, we depict the overall model architecture following the style in [34]. On the right, we expand the details of the generator and the discriminator components. The generator G generates an image conditioned on a synthetic image $\mathbf{x}^s$ and a noise vector $\mathbf{z}$. The discriminator D discriminates between real and fake images. The task–specific classifier T assigns task–specific labels $\mathbf{y}$ to an image. A convolution with stride 1 and 64 channels is indicated as n64s1 in the image. lrelu stands for leaky ReLU nonlinearity. BN stands for a batch normalization layer and FC for a fully connected layer. Note that we are not displaying the specifics of T as those are different for each task and decoupled from the domain adaptation process.

Domain Adaptation

Results

Figure 3. Visualization of our model's ability to generate samples when trained to adapt MNIST to MNIST-M. (a) Source images $\mathbf{x}^s$ from MNIST; (b) The samples adapted with our model $G(\mathbf{x}^s, \mathbf{z})$ with random noise $\mathbf{z}$; (c) The nearest neighbors in the MNIST-M training set of the generated samples in the middle row. Differences between the middle and bottom rows suggest that the model is not memorizing the target dataset.

Conclusion

- GANs are powerful tools that help to give the power of imagination to AI.
- Applications in media and fashion (Adobe, Amazon)
- Can they crack the unsupervised learning problem? (Research!)