Learning a Probabilistic Latent Space via 3D Generative-Adversarial Modeling

Jiajun Wu Chenkai Zhang
Tianfan Xue William t. Freeman Joshua B. Tenenbaum
MIT CSAIL

Presented by Brady Zhou

Outline

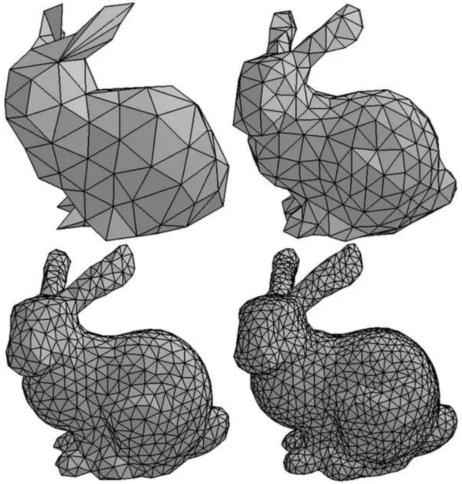
- Task
- Related Work
- Background
- Technical Details
- Experiments
- Discussion

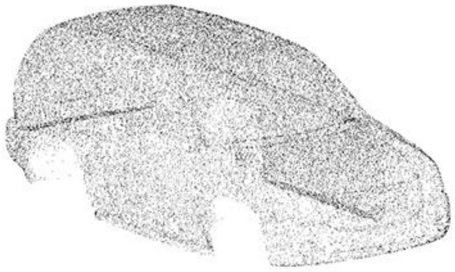
Outline

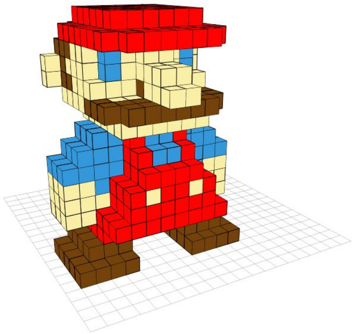
- **Task**
- Related Work
- Background
- Technical Details
- Experiments
- Discussion

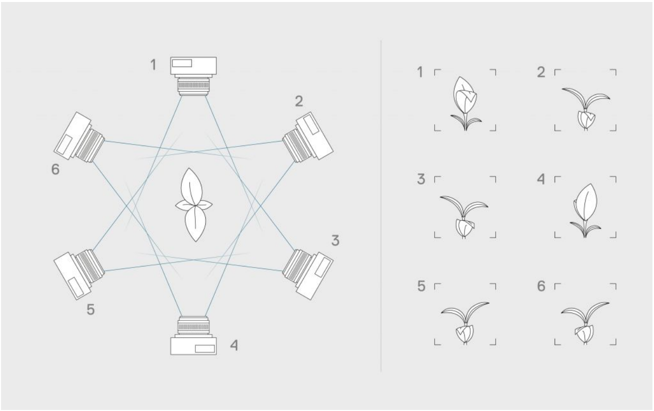
Task

- 3D Object Representation
  - Input: 3D object
    - Mesh
    - Point cloud
    - Multiple views
    - Implicit surface
  - Output: Feature vector
- Applications
  - Classification
  - Generation
  - Retrieval









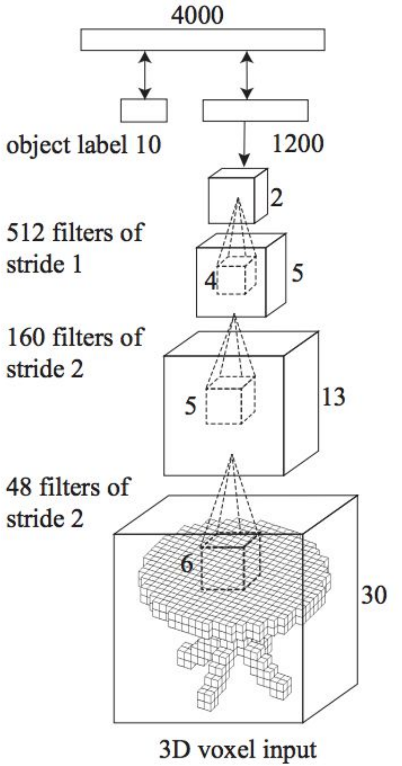
Outline

- Task
- **Related Work**
- Background
- Technical Details
- Experiments
- Discussion

Related Work

- 3D ShapeNets
  - Zhirong Wu, Shuran Song from Princeton
  - Deep belief network
  - Voxel representation (30x30x30)
  - ModelNet10/ModelNet40
  - 80.35%/77% classification

Figure 5: **ModelNet Dataset.** Left: word cloud visualization of the ModelNet dataset based on the number of 3D models in each category. Larger font size indicates more instances in the category. Right: Examples of 3D chair models.

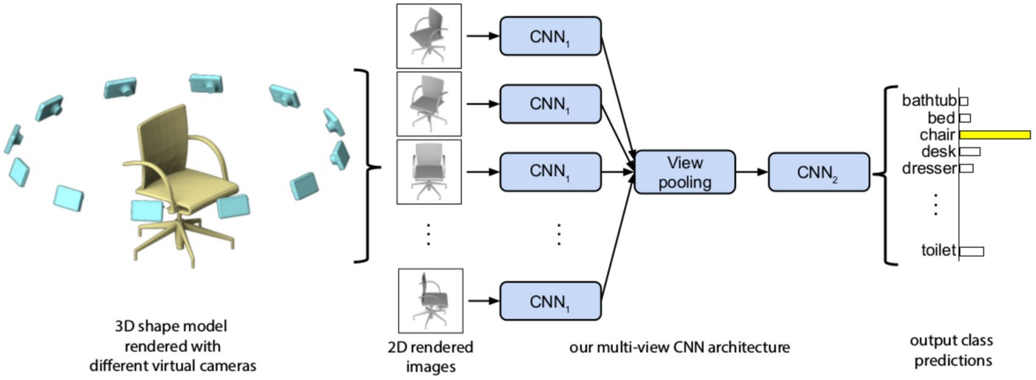


(a) Architecture of our 3D ShapeNets model. For illustration purpose, we only draw one filter for each convolutional layer.

[Z. Wu, S. Song]

Related Work

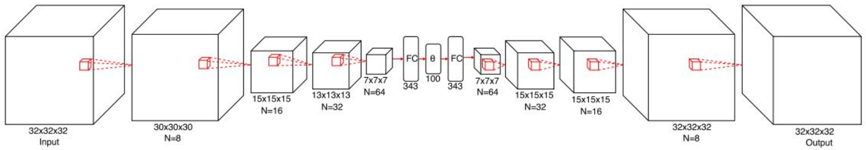
- MVCNN for 3D Shape Recognition
  - Hang Su, Subhransu Maji from UMass
  - 80 viewpoints
  - 2D representation
  - 90.1% classification on ModelNet40 (ImageNet pretrained)



[H. Su, S. Maji]

Related Work

- Generative and Discriminative Voxel Modeling with CNNs
  - Andrew Brock, Theodore Lim from Heriot-Watt University
  - Variational Autoencoder (VAE) Network
  - Voxel Representation (32x32x32)
  - State of the art
  - 97.14% on ModelNet10 classification
  - 95.54% on ModelNet40 classification



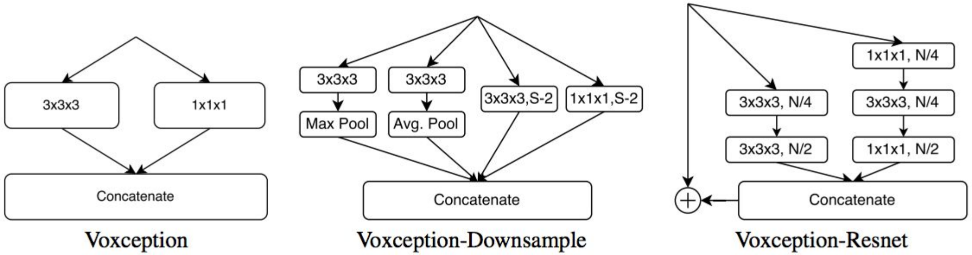
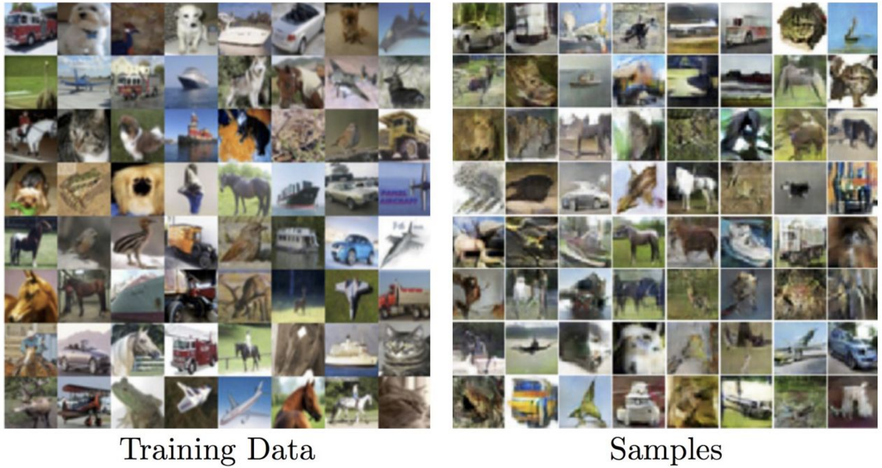


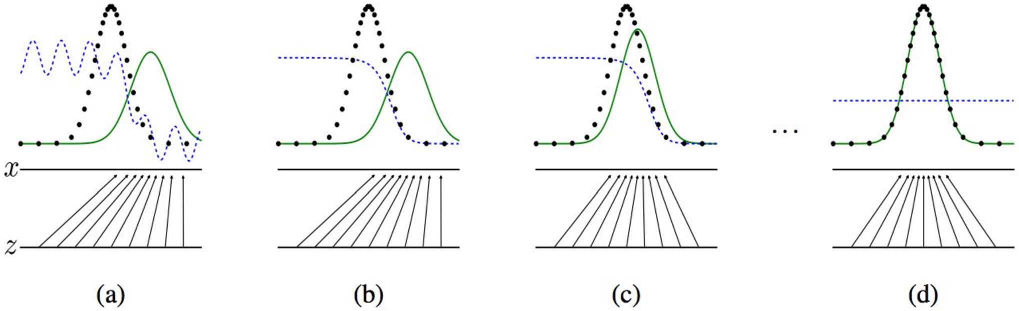
Figure 3: Voxception and Voxception-Resnet Blocks.

[A. Brock, T. Lim]

Related Work

- Generative Adversarial Networks
  - Ian Goodfellow, ‘14 NIPS
  - Generator/Discriminator
- Example:
  - Black - underlying data distribution
  - Green - generated data distribution
  - Blue - discriminator (separates true/generated)





[I. Goodfellow]

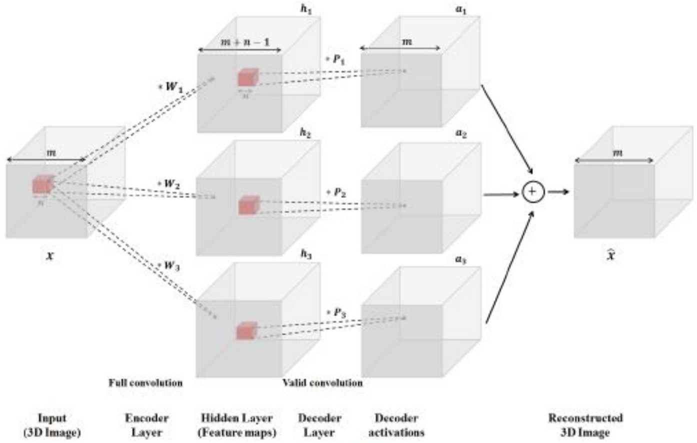
Outline

- Task
- Related Work
- **Background**
- Technical Details
- Experiments
- Discussion

Background

- 3D Convolution
  - Input: feature map (l x w x h x c), d filters of size (P x Q x R x c)
  - Output: feature map (l x w x h x d)
  - For the ith layer, the feature map position (x, y, z) from the jth filter

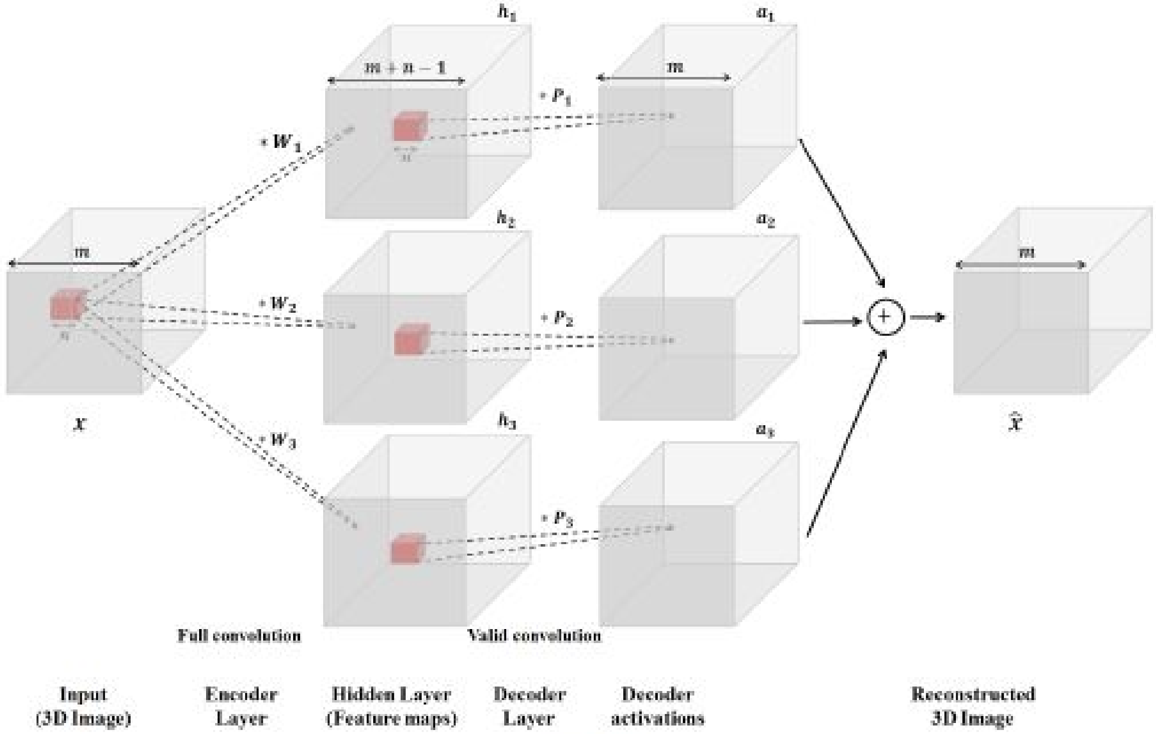
$$v_{ij}^{xyz} = \tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)}\right), \quad (2)$$



[S. Ji, W. Xu 3D CNN for Human Action Recognition]

Background

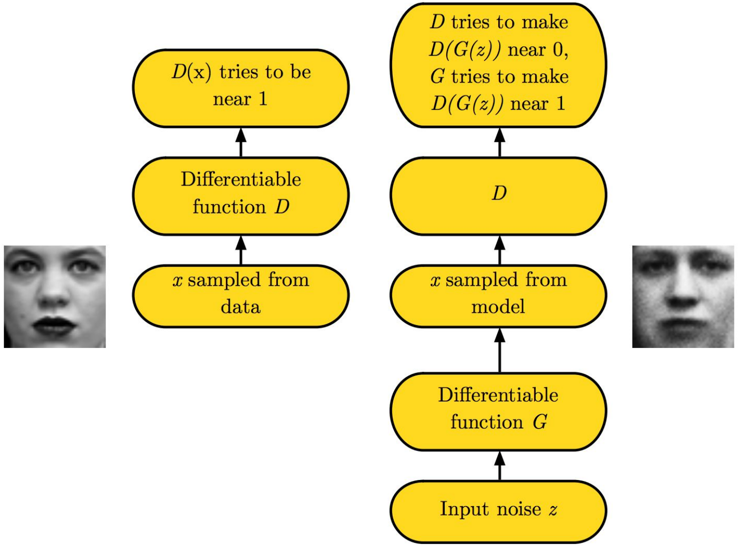
- 3D Convolution



[S. Ji, W. Xu 3D CNN for Human Action Recognition]

Background

- Generative Adversarial Network
  - Generator:
    - Input: latent variable (noise)
    - Output: reconstructed object
  - Discriminator:
    - Input: real or fake image
    - Output: probability of real

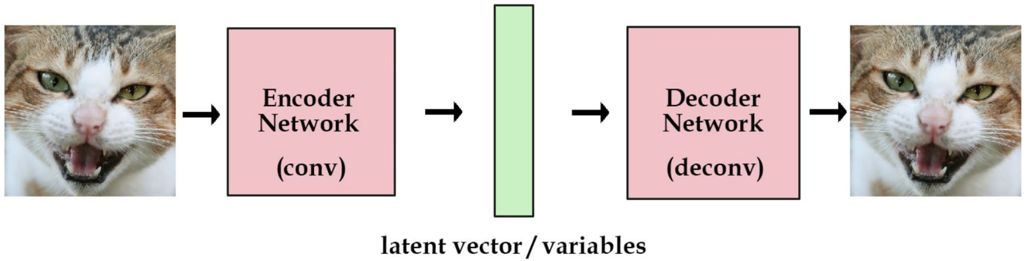


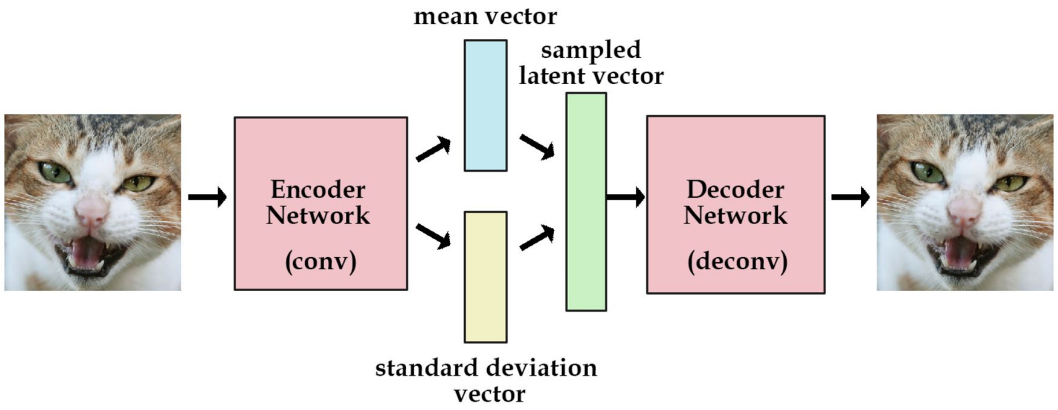
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

[Goodfellow ‘16 NIPS]

Background

- Autoencoder
  - Input: real image
  - Output: generated image
  - Loss: l2 difference
- Variational Autoencoder
  - Input: real image
  - Output: generated image
  - Loss: l2 difference + latent loss





[http://kvfrans.com/variational-autoencoders-explained/]

Background

- KL Divergence
  - Difference between probability distributions
  - Used as a latent loss

The KL divergence $D_{KL}(P_{fair} || P_{loaded})$ is just the difference between the cross-entropy $H(P_{fair}, P_{loaded})$ and the entropy of P_{fair}:

$$D_{KL}(P_{fair} || P_{loaded}) = H(P_{fair}, P_{loaded}) - H(P_{fair}) \cong 2.15. \qquad (6)$$

In other words, **the KL divergence $D_{KL}(P_{fair} || P_{loaded})$ measures "how much more surprised I expect you to be if I roll the fair dice but you mistakenly believe I'm rolling the loaded dice than if I roll the fair dice and you correctly believe I'm doing so."**

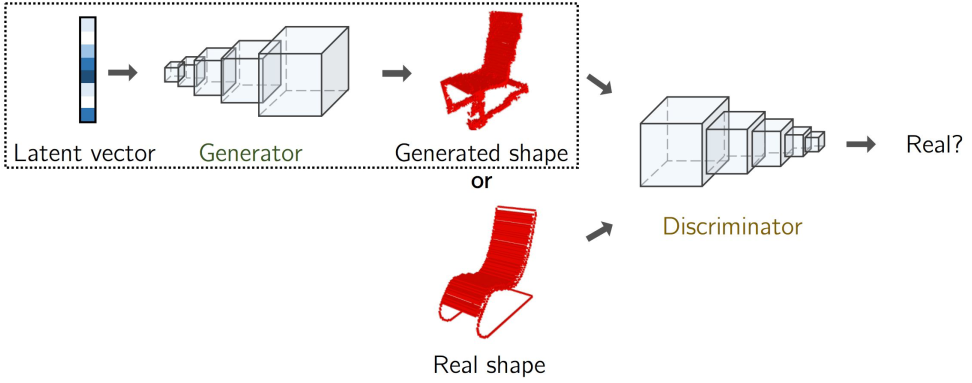
[https://www.quora.com/What-is-a-good-laymans-explanation-for-the-Kullback-Leibler-Divergence]

Outline

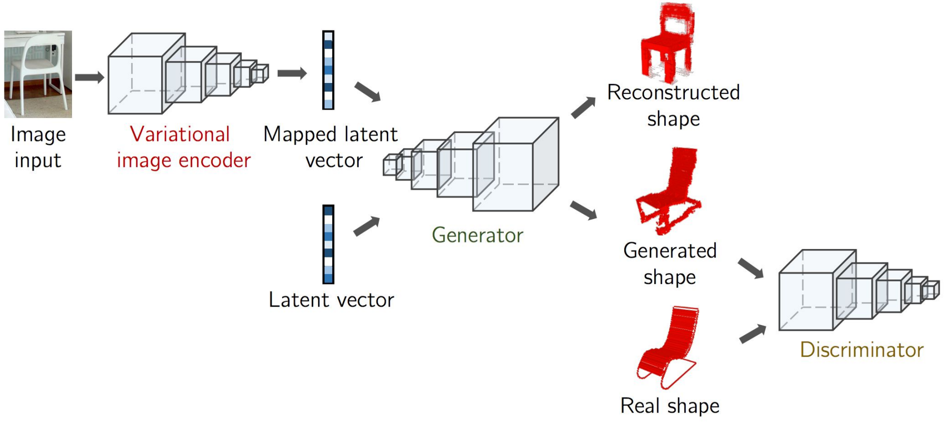
- Task
- Related Work
- Background
- **Technical Details**
- Experiments
- Discussion

Technical Details

- 3D-GAN



- 3D-VAE-GAN



[J. Wu, C. Zhang]

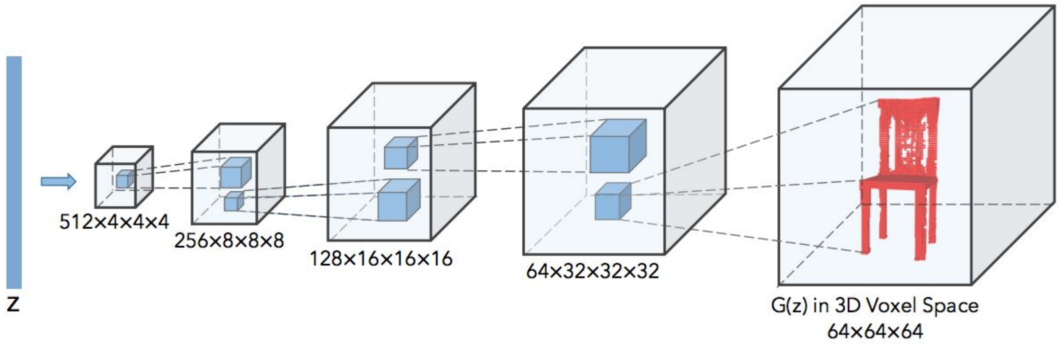
Technical Details

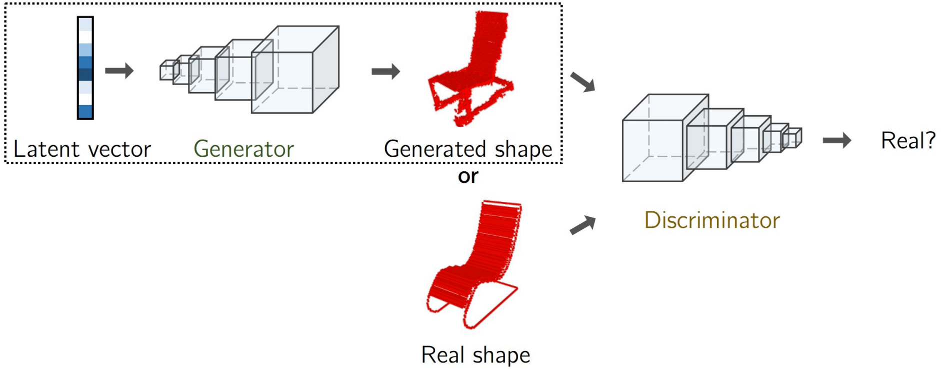
- Datasets
  - ModelNet10, ModelNet40
    - 3D model, class label pairs
    - 4941 3D models in ModelNet10
  - IKEA (J. Lim ‘13 ICCV)
    - Image, 3D model pairs
    - 759 images, 219 3D models
    - Real photos
    - Occlusion

[Z. Wu, J. Lim]

Technical Details

- 3D-GAN
  - Generator
    - Input: 200-d vector
    - Output: 64x64x64 grid
    - Kernel size {4, 4, 4, 4, 4}
    - Kernel stride {2, 2, 2, 2, 2}
    - Kernel # {512, 256, 128, 64, 1}
    - Relu, sigmoid at end
    - Batch normalization
  - Discriminator
    - Input: 64x64x64 grid
    - Output: probability of "real"
    - Mirrors discriminator
    - Leaky relu instead of relu



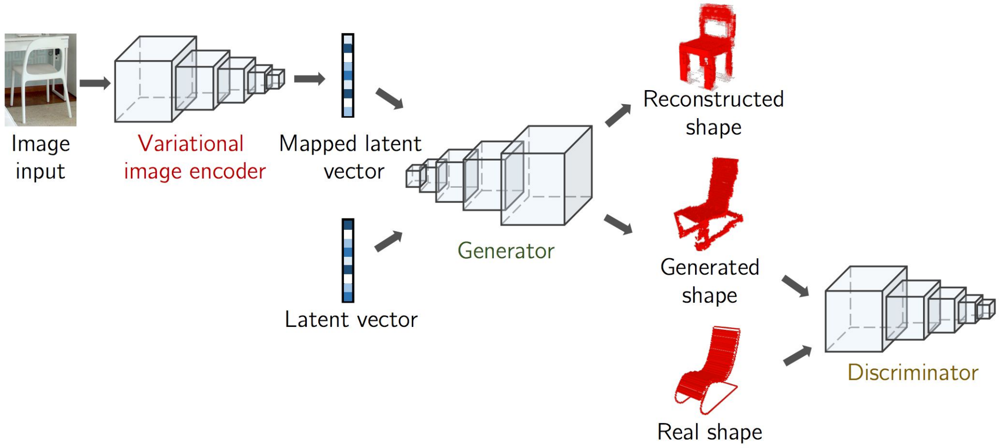


$$L_{\text{3D-GAN}} = \log D(x) + \log(1 - D(G(z))),$$

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE
  - Encoder
    - Input: image
    - Output: 200-d latent vector
    - Kernel size {11, 5, 5, 8}
    - Kernel stride {4, 2, 2, 1}
  - Notation
    - Image y
    - Latent vector z
    - “Real” 3D shape x
  - Losses
    - 3D-GAN (x or z)
    - KL (y, compared to z)
    - Recon (x, y) pairs



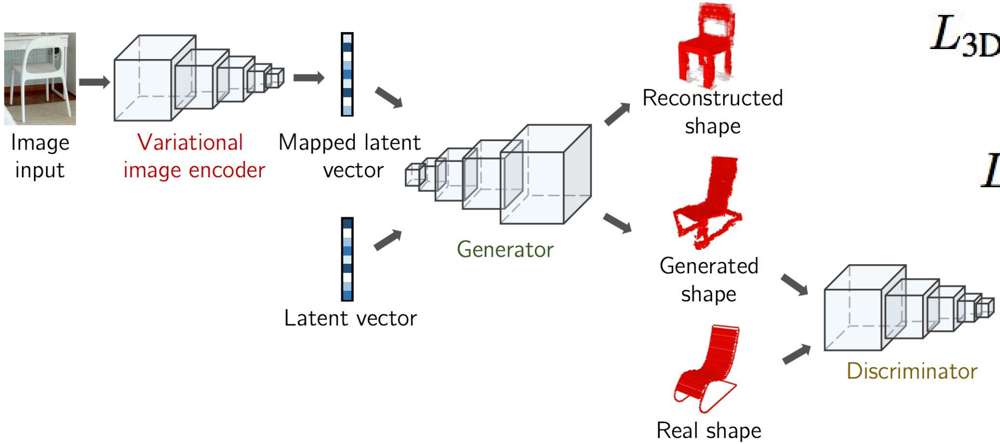
$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$L_{\text{3D-GAN}} = \log D(x) + \log(1 - D(G(z))),$$
$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$
$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE



- Notation
  - Image y
  - Latent vector z (Gaussian)
  - “Real” 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$L_{\text{3D-GAN}} = \log D(x) + \log(1 - D(G(z))),$$
$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$
$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$

[J. Wu, C. Zhang]

Technical Details

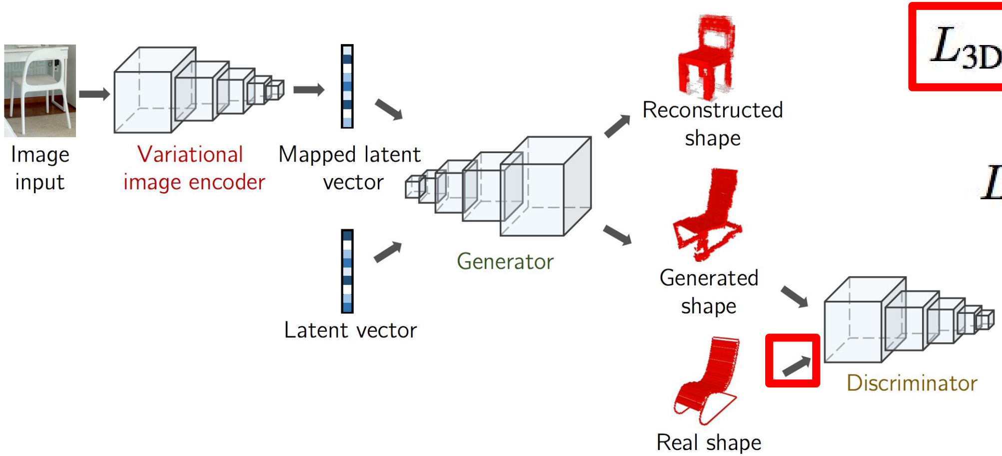
- 3D-GAN-VAE
  - Notation
    - Image y
    - Latent vector z (Gaussian)
    - "Real" 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$\boxed{L_{\text{3D-GAN}}} = \boxed{\log D(x)} + \log(1 - D(G(z))),$$

$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$

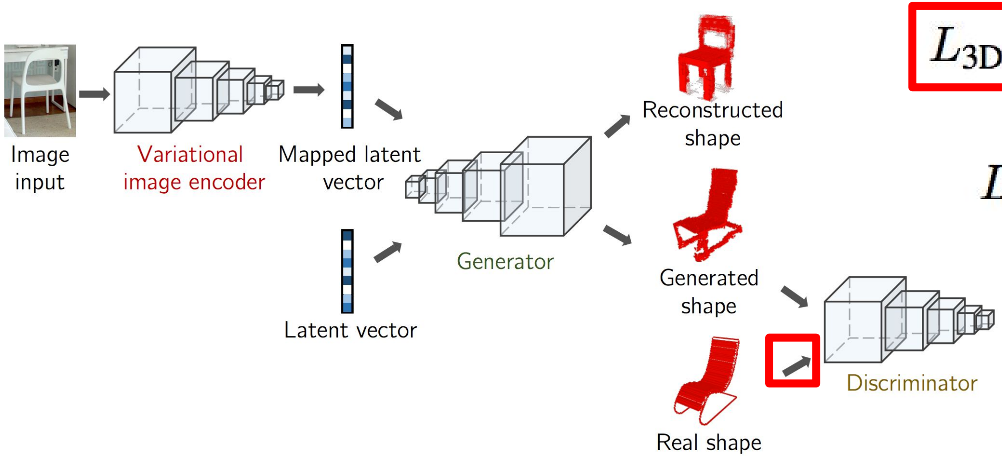
$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$



[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE



- Notation
  - Image y
  - Latent vector z (Gaussian)
  - “Real” 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$\boxed{L_{\text{3D-GAN}}} = \boxed{\log D(x)} + \log(1 - D(G(z))),$$

$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$

$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$

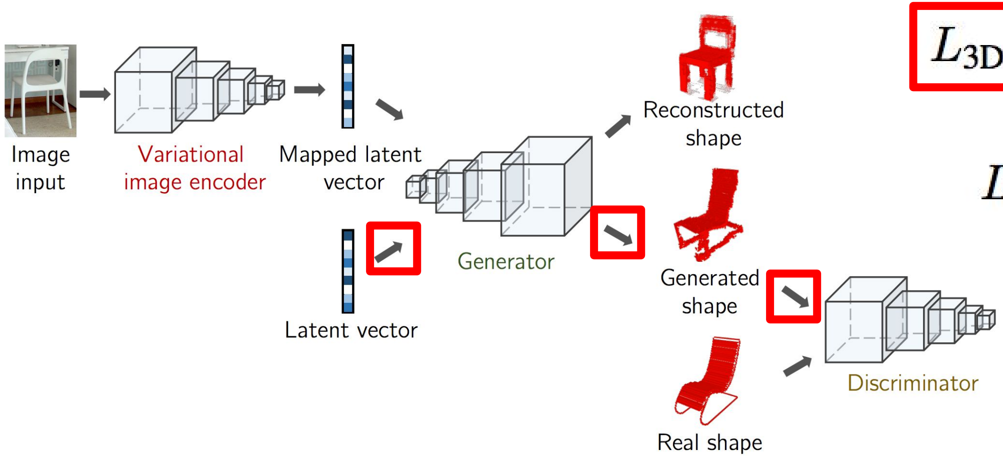
- Training D with real samples
  - D(x) = 0.9, log(0.9) = -0.045
  - D(x) = 0.1, log(0.1) = -0.1
  - Maximize loss

W <- W + h * dLoss

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE



- Notation
  - Image y
  - Latent vector z (Gaussian)
  - “Real” 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$\boxed{L_{\text{3D-GAN}}} = \log D(x) + \boxed{\log(1 - D(G(z)))},$$

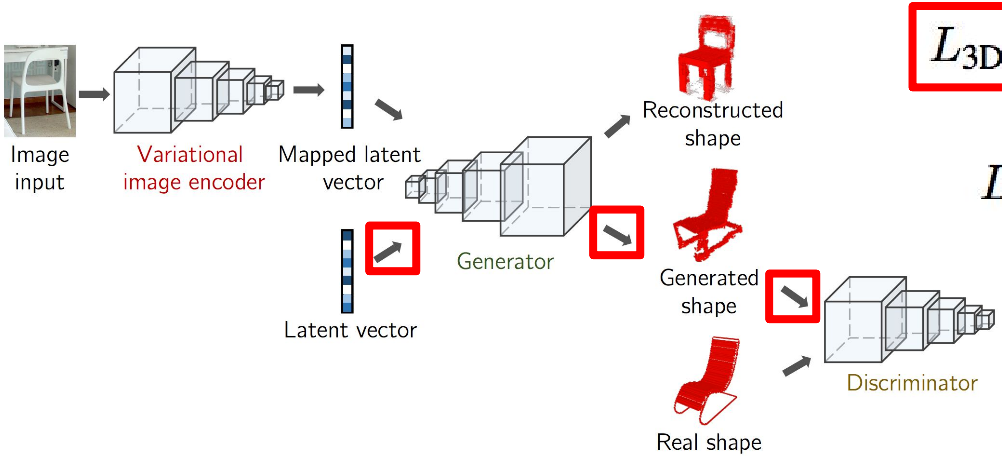
$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$

$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE



- Notation
  - Image y
  - Latent vector z (Gaussian)
  - “Real” 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$\boxed{L_{\text{3D-GAN}}} = \log D(x) + \boxed{\log(1 - D(G(z)))},$$

$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$

$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$

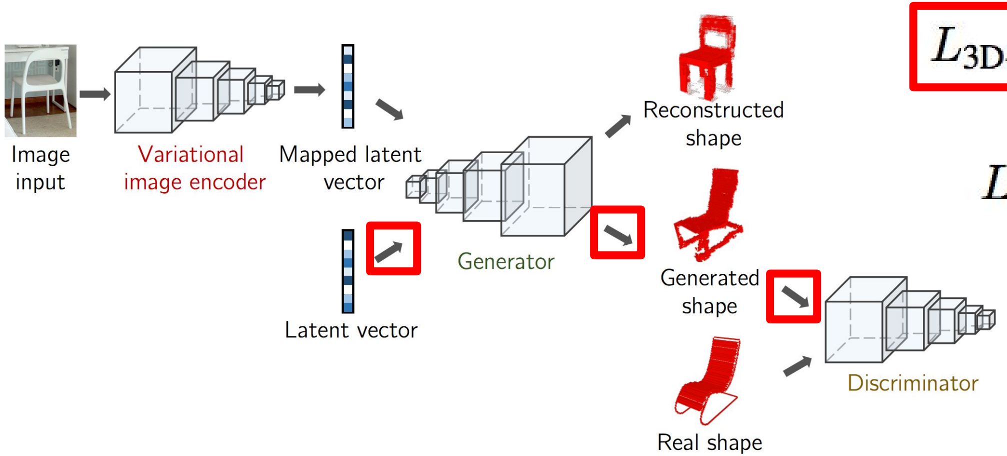
- Training D with G samples
  - D(G(z)) = 0.9, log(1-0.9) = -1
  - D(G(z)) = 0.1, log(1-0.1) = -0.045
  - Maximize loss

W <- W + h * dLoss

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE



W <- W + -h * dLoss

- Notation
  - Image y
  - Latent vector z (Gaussian)
  - “Real” 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$\boxed{L_{\text{3D-GAN}}} = \log D(x) + \boxed{\log(1 - D(G(z)))},$$

$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$

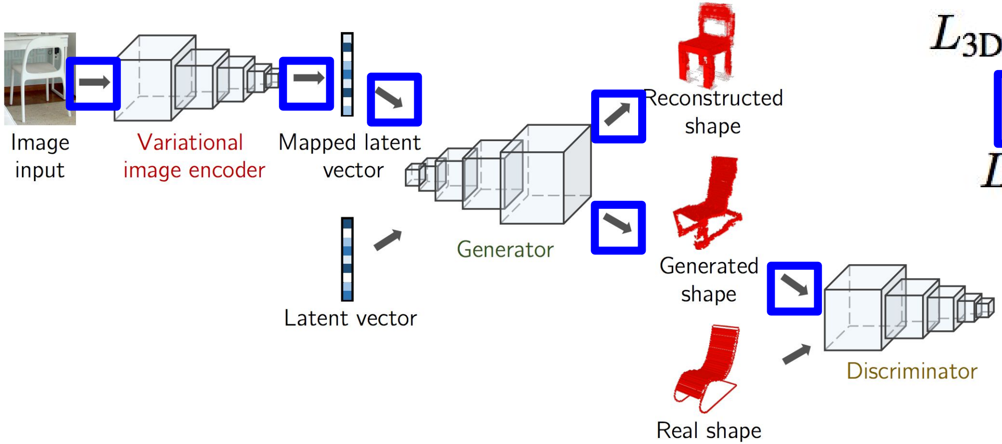
$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$

- Training G
  - D(G(z)) = 0.9, log(1-0.9) = -1
  - D(G(z)) = 0.1, log(1-0.1) = -0.045
  - Minimize loss

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE



W <- W + -h * dLoss

  - Notation
    - Image y
    - Latent vector z (Gaussian)
    - "Real" 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

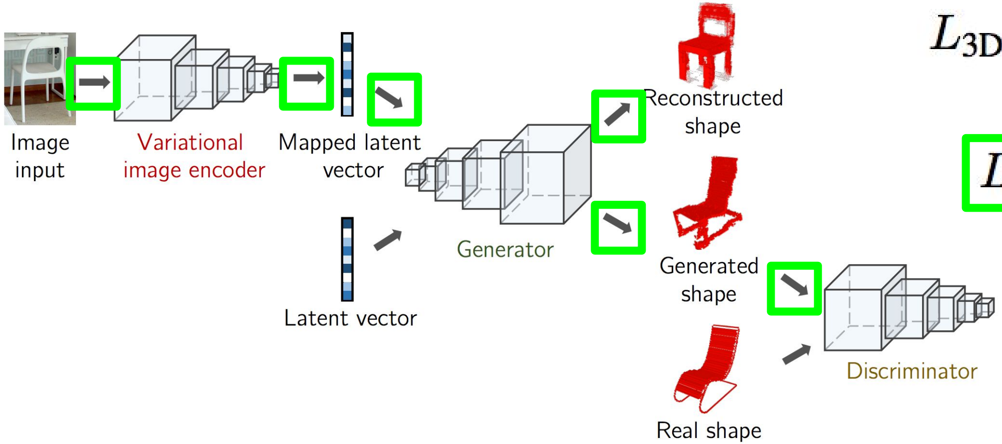
$$L_{\text{3D-GAN}} = \log D(x) + \log(1 - D(G(z))),$$
$$\boxed{L_{\text{KL}}} = D_{\text{KL}}(q(z|y) \mid\mid p(z)),$$
$$L_{\text{recon}} = ||G(E(y)) - x||_2,$$

- Training E
  - VAE outputs mean, std vectors
  - Compare to Gaussian

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN-VAE



W <- W + -h * dLoss

- Notation
  - Image y
  - Latent vector z (Gaussian)
  - “Real” 3D shape x

$$L = L_{\text{3D-GAN}} + \alpha_1 L_{\text{KL}} + \alpha_2 L_{\text{recon}},$$

$$L_{\text{3D-GAN}} = \log D(x) + \log(1 - D(G(z))),$$

$$L_{\text{KL}} = D_{\text{KL}}(q(z|y) \,||\, p(z)),$$

$$\boxed{L_{\text{recon}}} = ||G(E(y)) - x||_2,$$

- Training E
  - VAE outputs reconstructed
  - Compare to actual

[J. Wu, C. Zhang]

Technical Details

- 3D-GAN Training
  - Update discriminator if last batch accuracy < 80%
  - Generator learning rate 0.0025
  - Discriminator learning rate 1e-5
  - Batch size 100
  - ADAM with beta 0.5

Outline

- Task
- Related Work
- Background
- Technical Details
- **Experiments**
- Discussion

Experiments

- Classification
  - Concatenate conv layers 2, 3, 4
  - Max pool with stride 8, 4, 2
  - Linear SVM

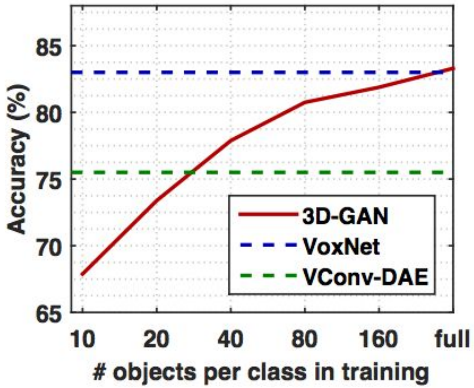


Figure 4: ModelNet40 classification with limited training data

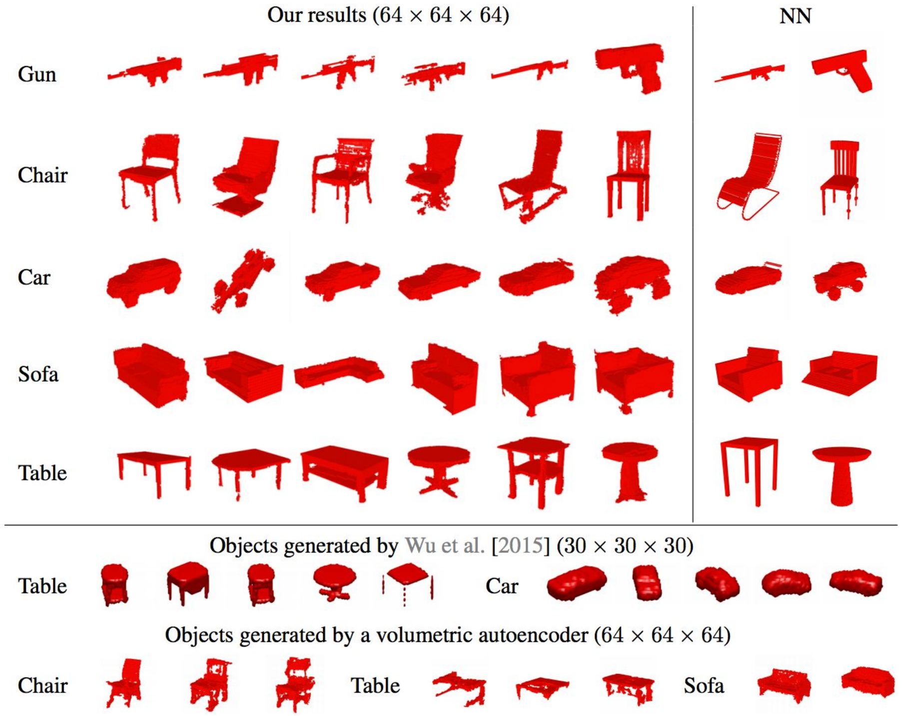
Supervision	Pretraining	Method	Classification (Accuracy)	
			ModelNet40	ModelNet10
Category labels	ImageNet	MVCNN [Su et al., 2015a]	90.1%	-
		MVCNN-MultiRes [Qi et al., 2016]	**91.4%**	-
	None	3D ShapeNets [Wu et al., 2015]	77.3%	83.5%
		DeepPano [Shi et al., 2015]	77.6%	85.5%
		VoxNet [Maturana and 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%**

Table 1: Classification results on the ModelNet dataset. Our 3D-GAN outperforms other unsupervised learning methods by a large margin, and is comparable to some recent supervised learning frameworks.

[J. Wu, C. Zhang]

Experiments

- Generation
  - G(z), randomly sampled 200-d z
  - Render largest connect component
  - Retrieve nearest neighbors
    - Use last conv layer



[J. Wu, C. Zhang]

Experiments

- Single Image 3D Reconstruction
  - Input: image
  - Output: 3D model

Method	Bed	Bookcase	Chair	Desk	Sofa	Table	Mean
AlexNet-fc8 [Girdhar et al., 2016]	29.5	17.3	20.4	19.7	38.8	16.0	23.6
AlexNet-conv4 [Girdhar et al., 2016]	38.2	26.6	31.4	26.6	69.3	19.1	35.2
T-L Network [Girdhar et al., 2016]	56.3	30.2	32.9	25.8	71.7	23.3	40.0
3D-VAE-GAN (jointly trained)	49.1	31.9	42.6	34.8	**79.8**	33.1	45.2
3D-VAE-GAN (separately trained)	**63.2**	**46.3**	**47.2**	**40.7**	78.8	**42.3**	**53.1**

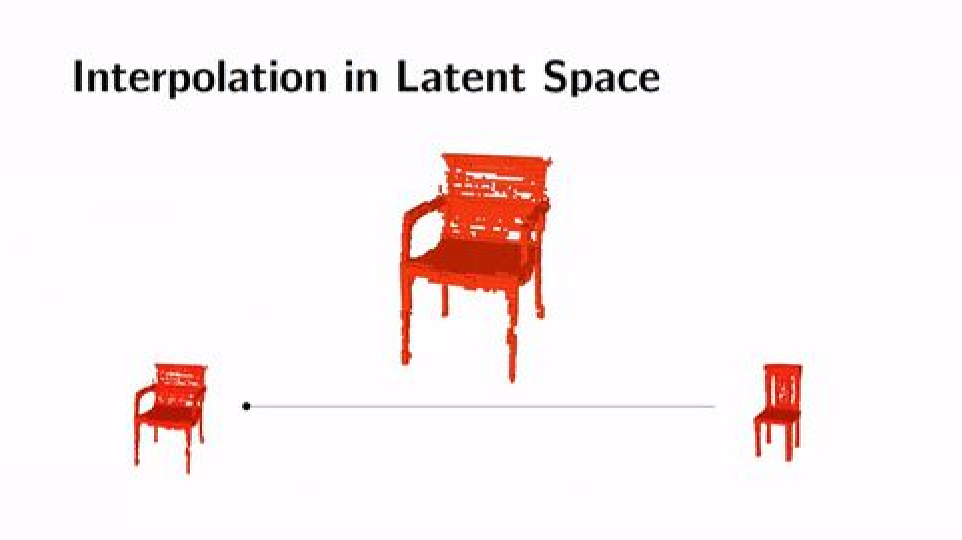
Table 2: Average precision for voxel prediction on the IKEA dataset.†

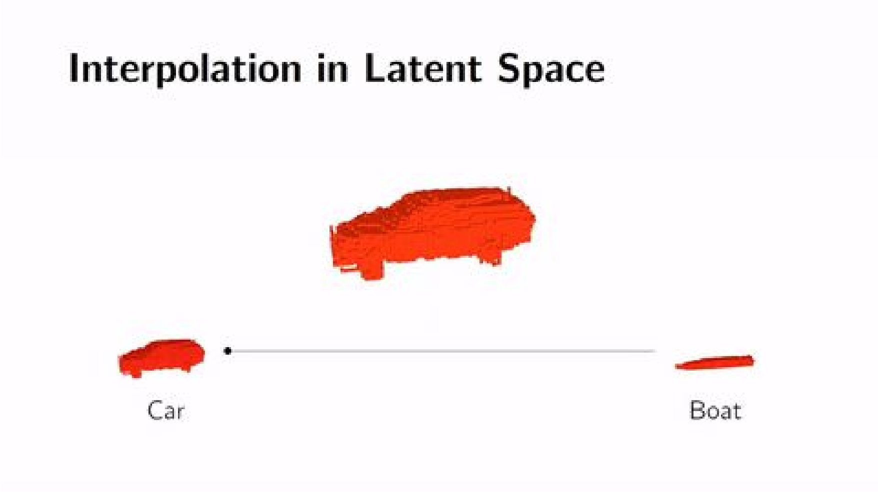
Figure 7: Qualitative results of single image 3D reconstruction on the IKEA dataset

[J. Wu, C. Zhang]

Experiments

- Latent Space Interpolation
  - Input:
    - “Start” latent vector, u
    - “End” latent vector, v
    - z = (1 - h) v + h v

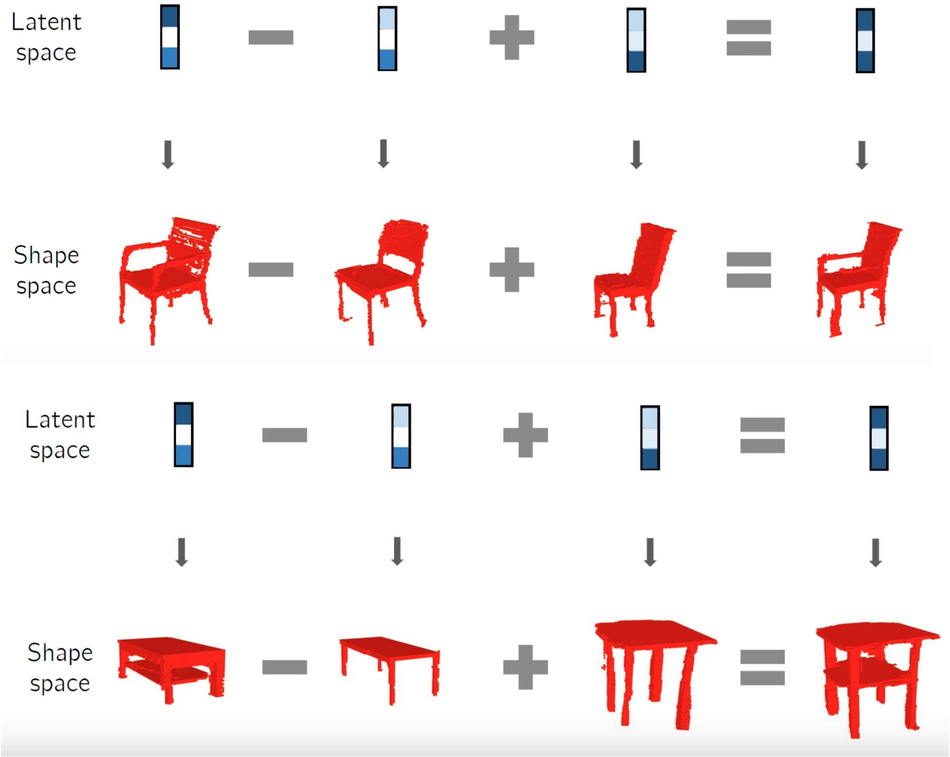




[J. Wu, C. Zhang]

Experiments

- Latent Space Arithmetic
  - Word2Vec
    - King - man + woman = queen



[J. Wu, C. Zhang]

Outline

- Task
- Related Work
- Background
- Technical Details
- Experiments
- **Discussion**

Discussion

- The Good
  - Simple pipeline
  - Unsupervised
- The Bad
  - Upper bound due to voxel representation

Thanks!

Questions?