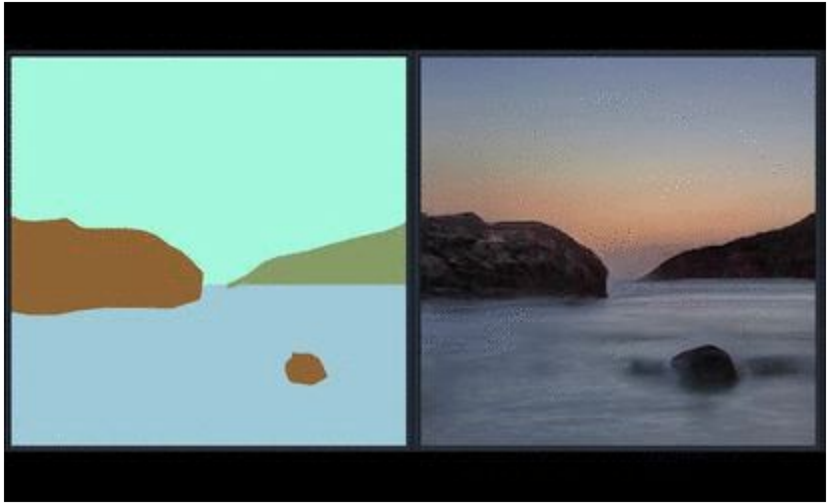
Semantic Image Synthesis with Spatially-Adaptive Normalization

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, Jun-Yan Zhu

Presented by - Ayushi Bansal, Bhargav Sundararajan, Mridula Gupta

Problem - Semantic Image Synthesis

Generate photorealistic image based on given input semantic layout

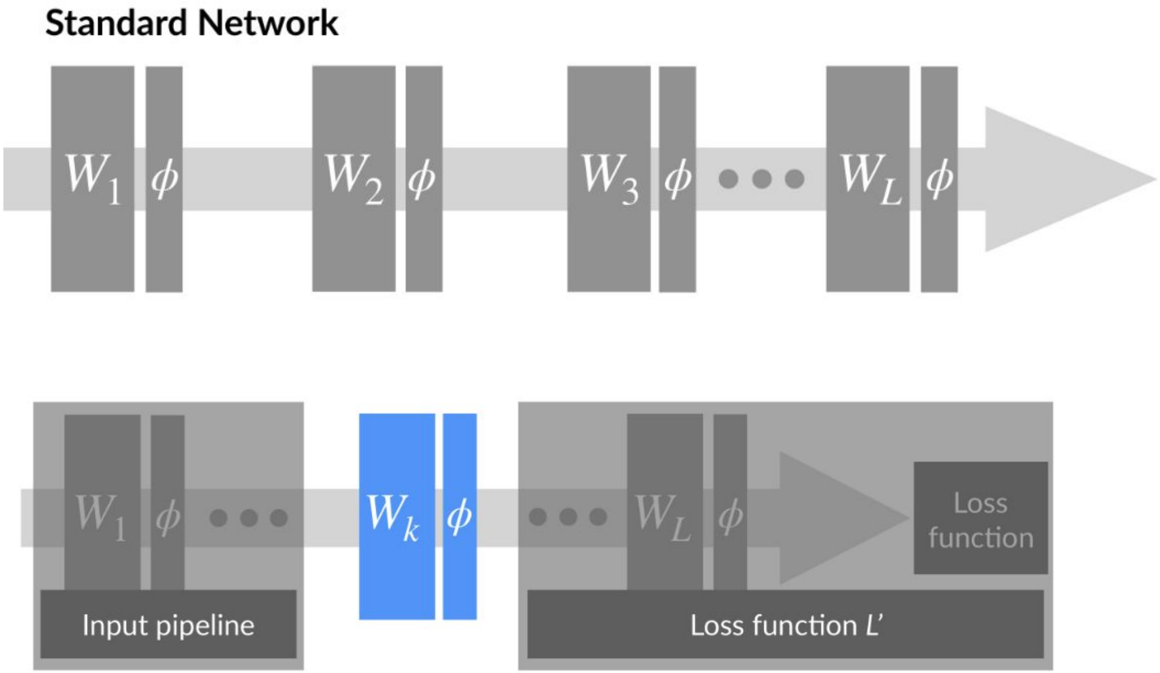


Normalization

- Adjusting values measured on different scales to a notionally common scale.
- Done by subtracting the mean and dividing by the standard deviation.
- Usually a data preprocessing step.

Normalization

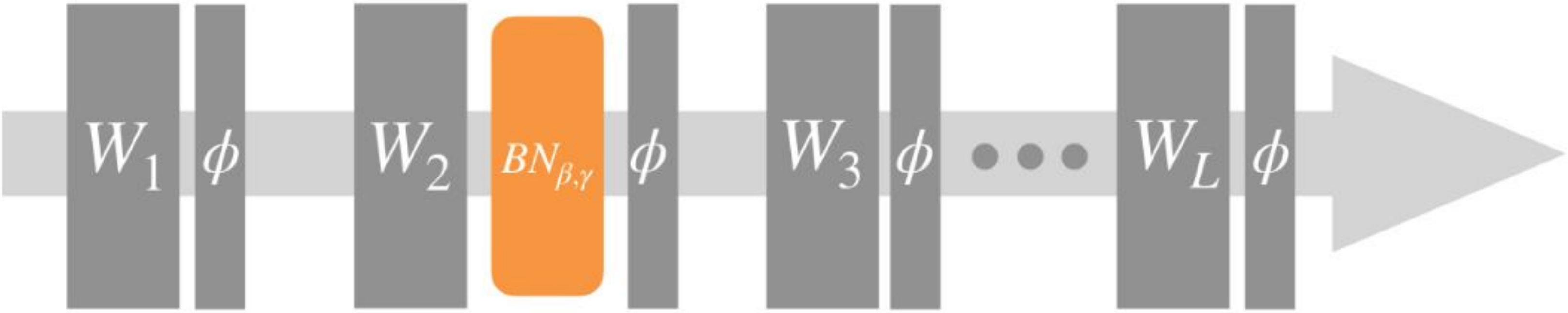
It is basically used to adjust and scale activations in neural network



http://gradientscience.org/batchnorm/

Normalization (Batch Norm)

Adding a BatchNorm layer (between weights and activation function)



Mean

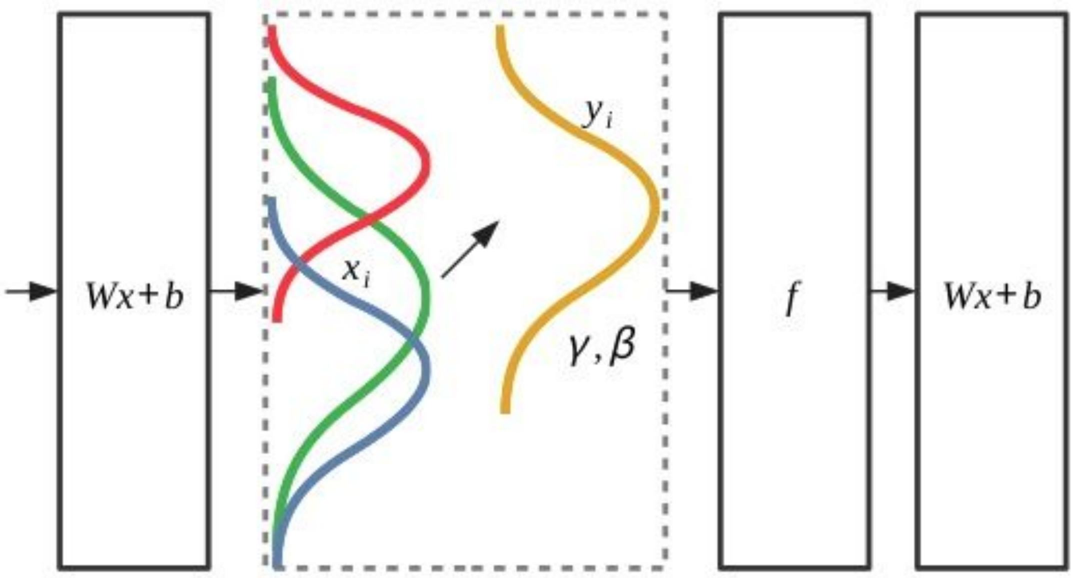
$$\overline{x_i} = \frac{x_i - \boxed{\mu_B}}{\sqrt{\boxed{\sigma_B^2} + \epsilon}}$$

Variance

Trainable Parameters (Scale and Shift)

$$y_i = \boxed{\gamma}\,\overline{x_i} + \boxed{\beta}$$

Why Normalization between layers?



Ensures that output statistics of the layer are fixed

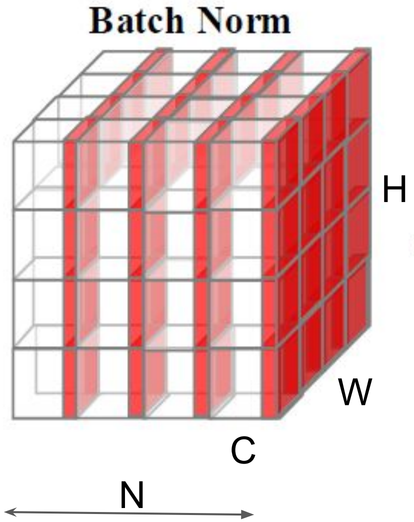
Unconditional Normalization

- No dependence on external data during normalization process
- Types
  - Batch Normalization
  - Instance Normalization
  - Layer Normalization
  - Group Normalization
  - Weight Normalization

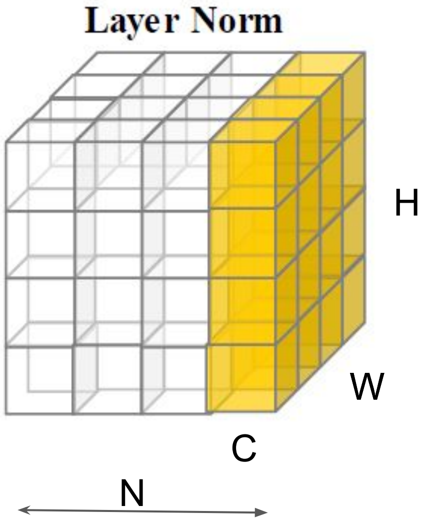
http://luoping.me/post/family-normalization/

Unconditional Normalization

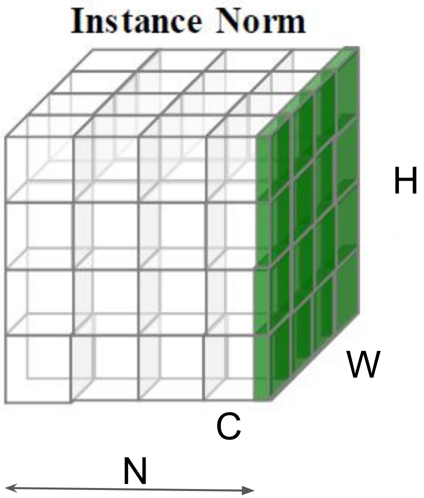
H	Height of Image
W	Width of Image
C	Channels
N	Number of Instances in a batch



Single Channel over Multiple training instances



Multiple Channel over single training instance



Single Channel over single training instance

Conditional Normalization

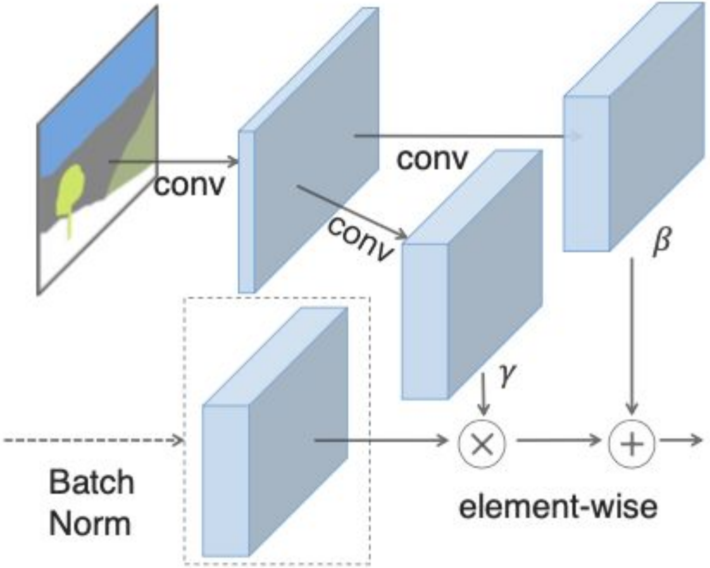
- External data is used to condition the normalization.
- Used in Style transfer and Visual Question Answering
- Example: Conditional Instance Normalization and Adaptive Instance Normalization

$$\mathrm{CIN}(x; s) = \gamma^s \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta^s$$

$$\mathrm{AdaIN}(x, y) = \sigma(y) \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \mu(y)$$

SPADE Model - Conditional Normalization

Unlike prior conditional normalization methods, γ and β are not vectors, but tensors with spatial dimensions. Hence, the name Spatially Adaptive Normalization



SPADE Model - Conditional Normalization

Previous Activation

Activation Value = $$\gamma^i_{c,y,x}(\mathbf{m})\frac{h^i_{n,c,y,x}-\mu^i_c}{\sigma^i_c}+\beta^i_{c,y,x}(\mathbf{m})$$

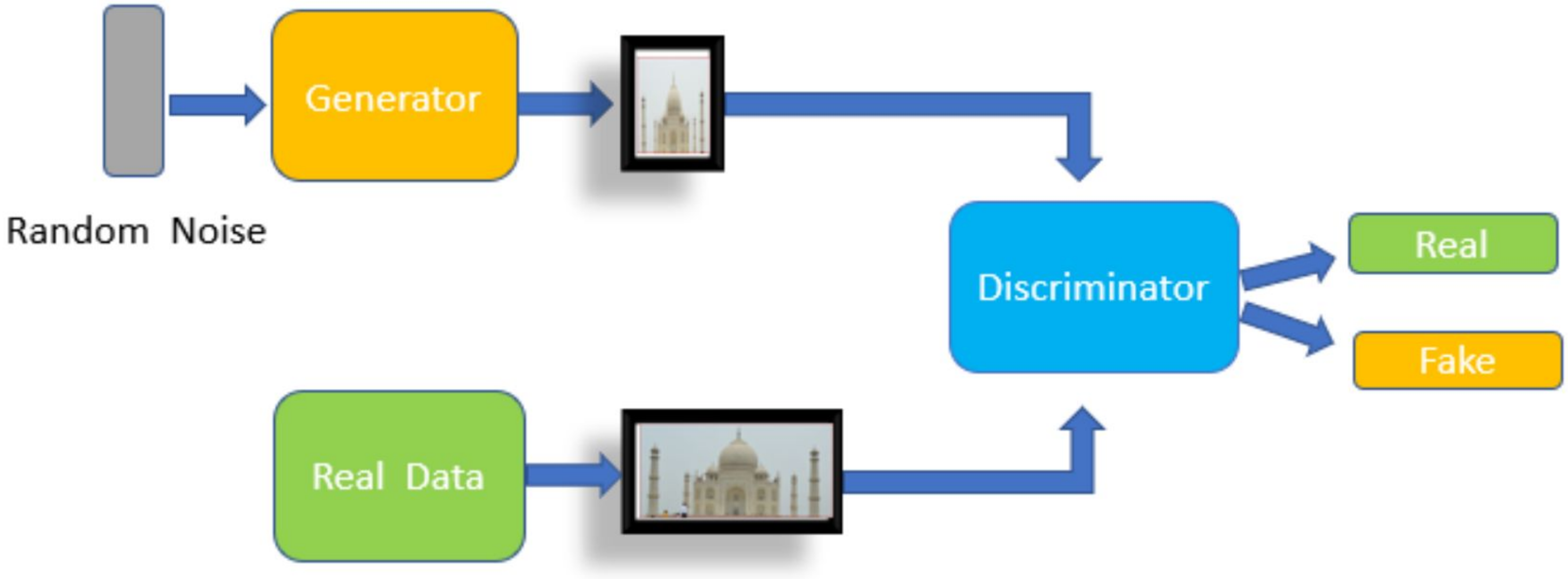
(n ∈ N, c ∈ Ci, y ∈ Hi, x ∈ Wi)

Learned modulation parameters

Mean $$\mu^i_c = \frac{1}{NH^iW^i}\sum_{n,y,x} h^i_{n,c,y,x}$$

SD $$\sigma^i_c = \sqrt{\frac{1}{NH^iW^i}\sum_{n,y,x}(h^i_{n,c,y,x})^2-(\mu^i_c)^2}.$$

GANs (Generative Adaptive Networks)



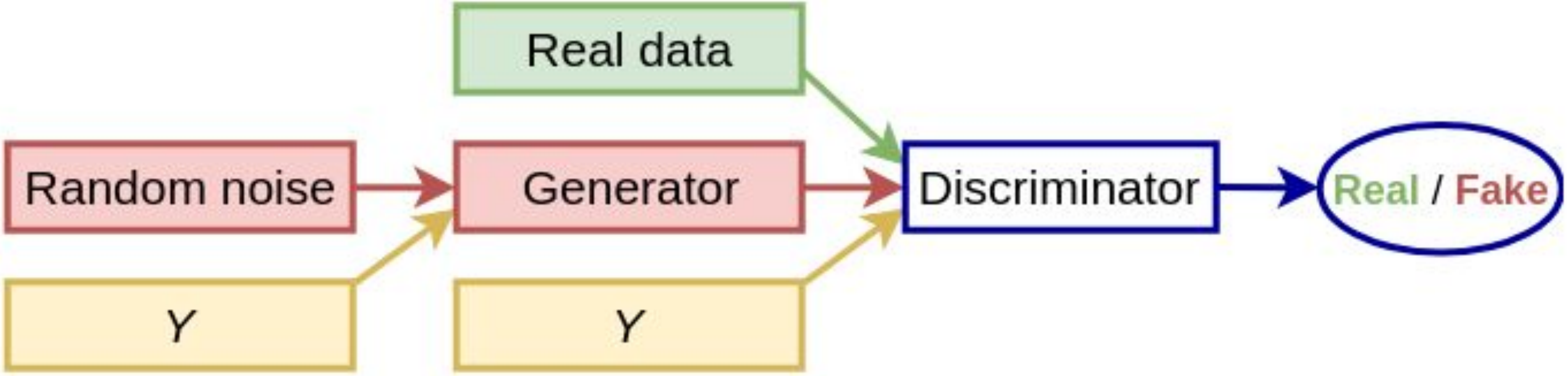
https://developers.google.com/machine-learning/gan/gan_structure

GANs (Generative Adaptive Networks)

- **Generator:** Learns to generates plausible image
- **Discriminator:** Learns to distinguish generator’s fake image from real image
- **Training**
  - Generator and Discriminator alternatively trained (1 epoch each)
  - Converge the results

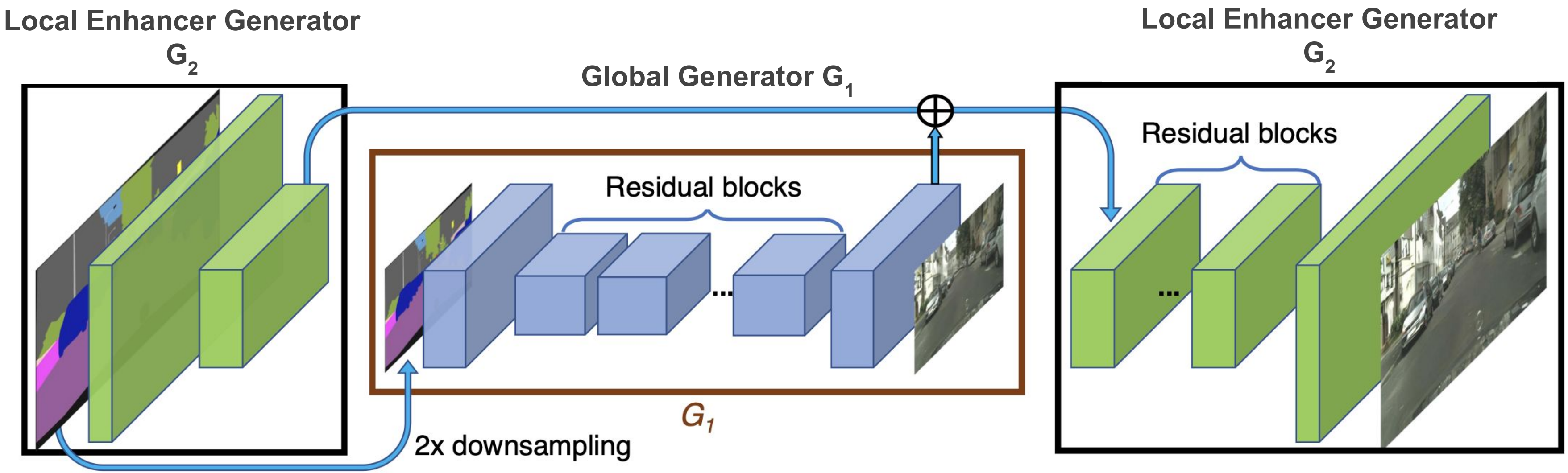
Conditional GANs

In addition to random noise (for generator) and input image (for discriminator), we are giving **one-hot encoded vector** as input (Y)



https://medium.com/@connorshorten300/conditional-gans-639aa3785122

Pix2PixHD Model



Training

- Train global generator G_1 separately first
- Fine-tune whole network with G_1 and G_2

Pix2PixHD Model

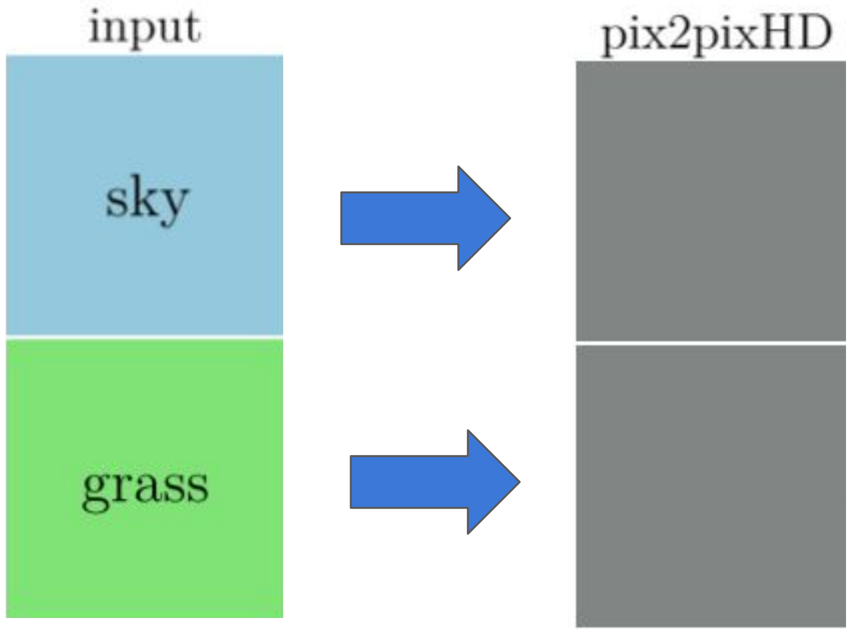
Two main contributions:

1. Instance-level object segmentation information is used, which can separate different object instances within the same category.

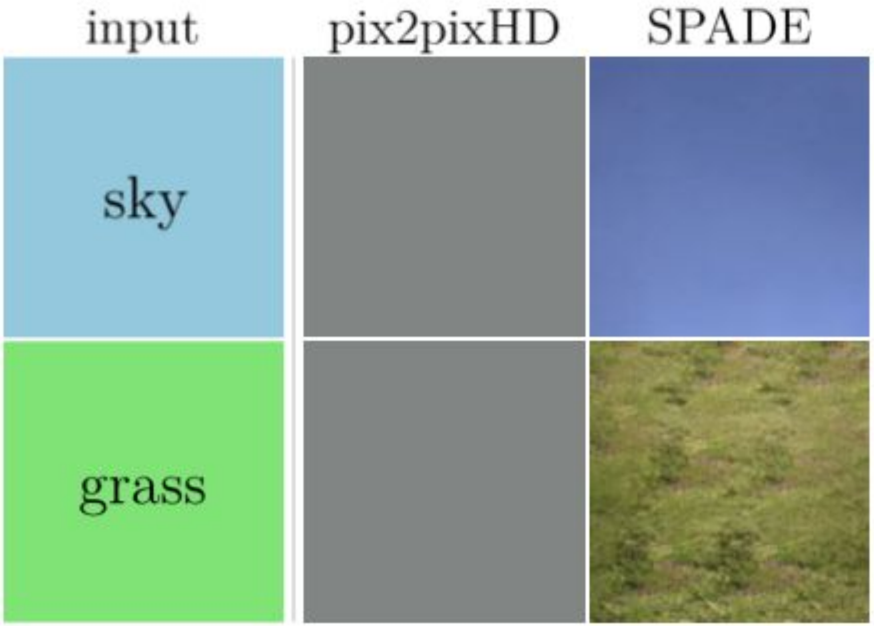
2. Generate diverse results given the same input label map, allowing the user to edit the appearance of the same object interactively.

Motivation - Pix2PixHD

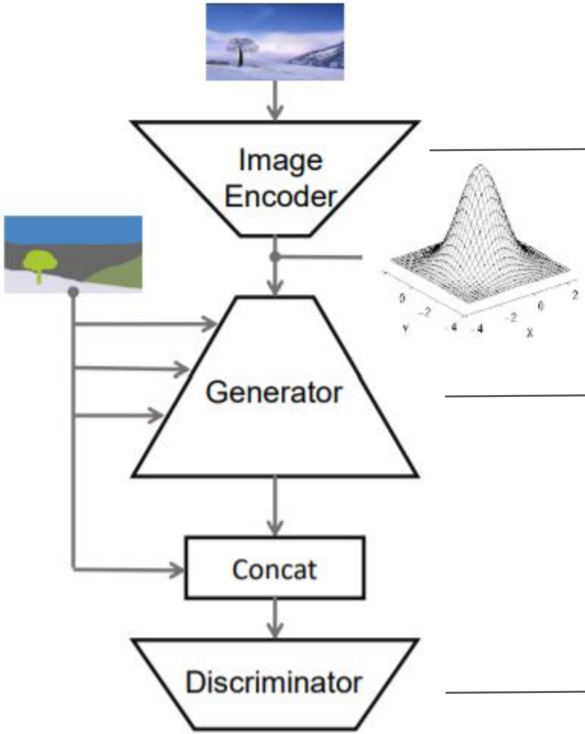
Normalization “washes” away semantic information when applied to uniform or flat segmentation mask



Resolves Pix2PixHD problem



The Model Architecture

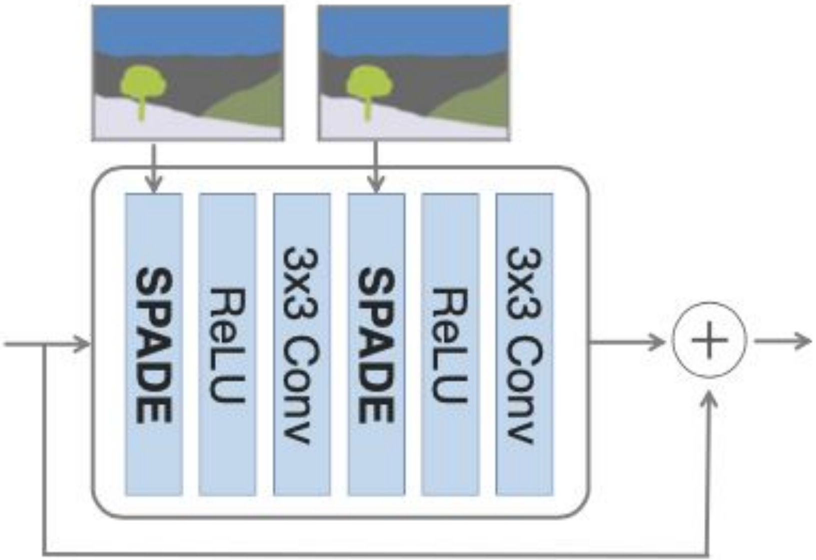


Encodes real image to mean vector and variance vector

Generates images based on noise distribution

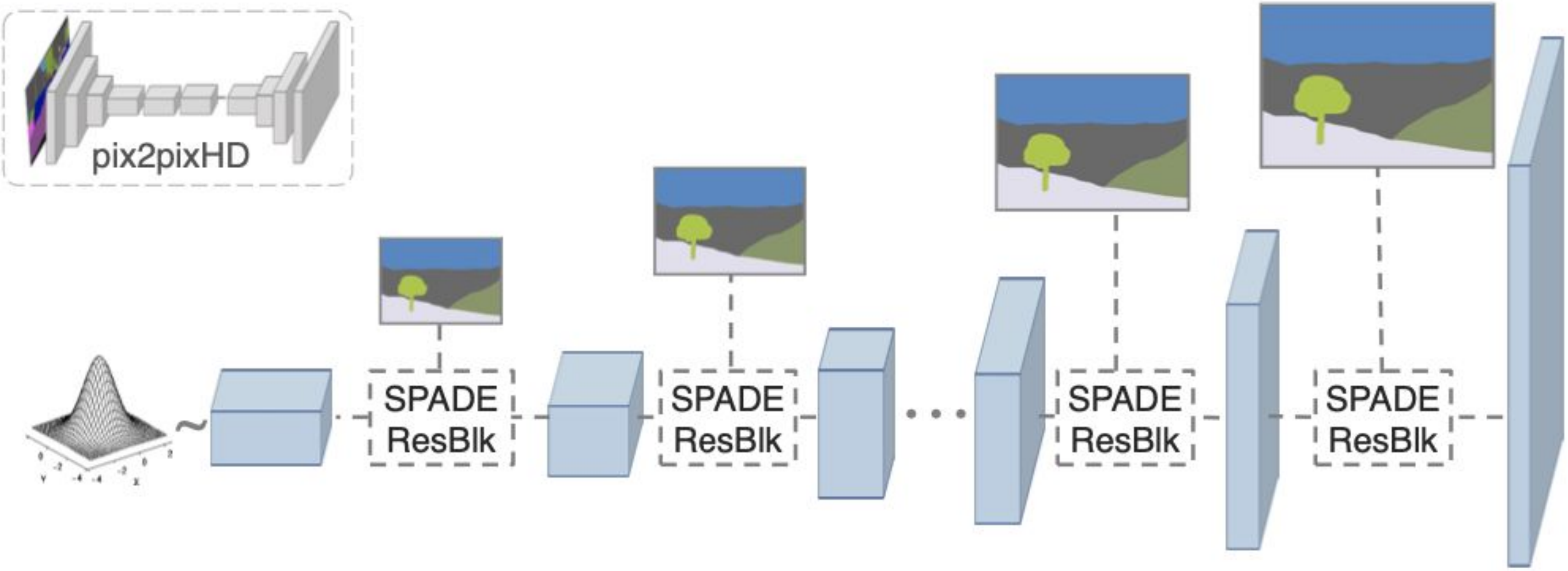
Discriminates synthesized images from ground truth and propagates loss

SPADE Model - Residual Block

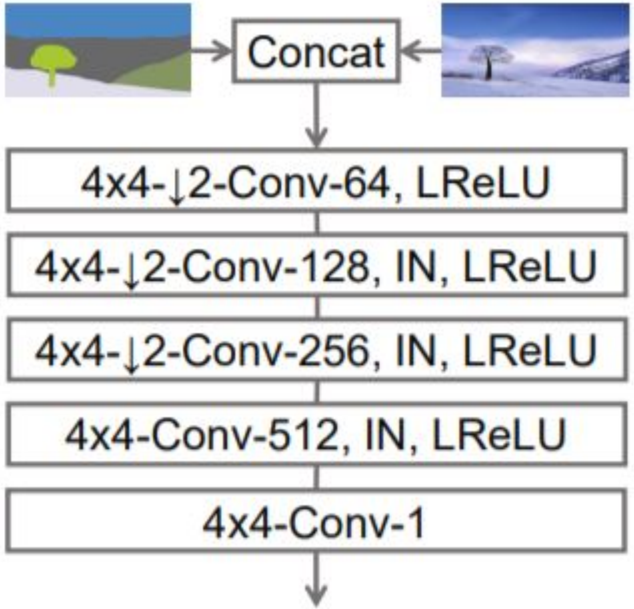


SPADE ResBlk

SPADE Model - Generator



Discriminator



Loss Function

L_{FM} compares the feature map of every intermediate layer of the discriminator.

$$\min_{G}\left(\left(\max_{D_1,D_2,D_3}\sum_{k=1,2,3}\mathcal{L}_{\mathrm{GAN}}(G,D_k)\right)+\lambda\sum_{k=1,2,3}\mathcal{L}_{\mathrm{FM}}(G,D_k)\right)$$

$$\mathcal{L}_{\mathrm{FM}}(G,D_k)=\mathbb{E}_{(\mathbf{s},\mathbf{x})}\sum_{i=1}^{T}\frac{1}{\boxed{N_i}}\left[||D_k^{(i)}(\mathbf{s},\mathbf{x})-D_k^{(i)}(\mathbf{s},G(\mathbf{s}))||_1\right],$$

Number of
Intermediate Layers

Loss - Hinge Loss

Prediction for
real Image

$$V_D(\hat{G}, D) = \boxed{\underset{\boldsymbol{x}\sim q_{\text{data}}(\boldsymbol{x})}{\mathrm{E}} \left[\min\left(0, -1 + D(\boldsymbol{x})\right)\right]} + \underset{\boldsymbol{z}\sim p(\boldsymbol{z})}{\mathrm{E}} \left[\min\left(0, -1 - D\left(\hat{G}(\boldsymbol{z})\right)\right)\right]$$

$$V_G(G, \hat{D}) = - \underset{\boldsymbol{z}\sim p(\boldsymbol{z})}{\mathrm{E}} \left[\hat{D}\left(G(\boldsymbol{z})\right)\right],$$

Loss - Hinge Loss

Prediction for
Generated Image

$$V_D(\hat{G}, D) = \underset{\boldsymbol{x} \sim q_{\text{data}}(\boldsymbol{x})}{\mathrm{E}} \left[\min\left(0, -1 + D(\boldsymbol{x})\right)\right] + \boxed{\underset{\boldsymbol{z} \sim p(\boldsymbol{z})}{\mathrm{E}} \left[\min\left(0, -1 - D\left(\hat{G}(\boldsymbol{z})\right)\right)\right]}$$

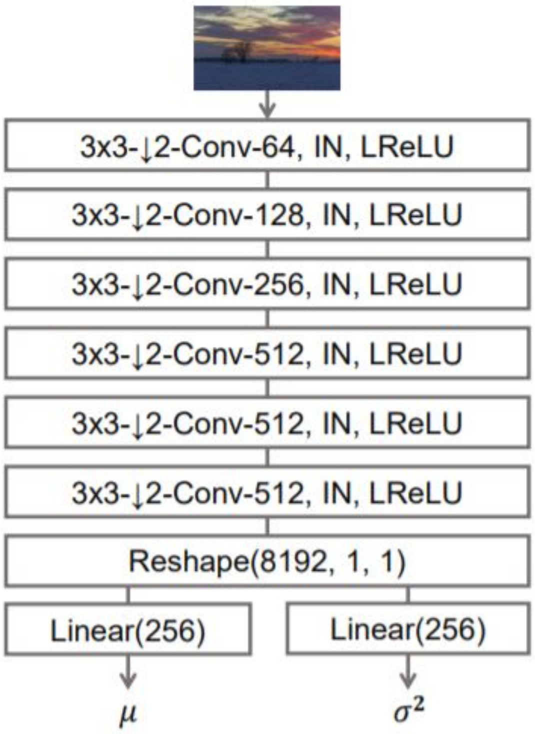
$$V_G(G, \hat{D}) = - \underset{\boldsymbol{z} \sim p(\boldsymbol{z})}{\mathrm{E}} \left[\hat{D}\left(G(\boldsymbol{z})\right)\right],$$

Loss - Hinge Loss

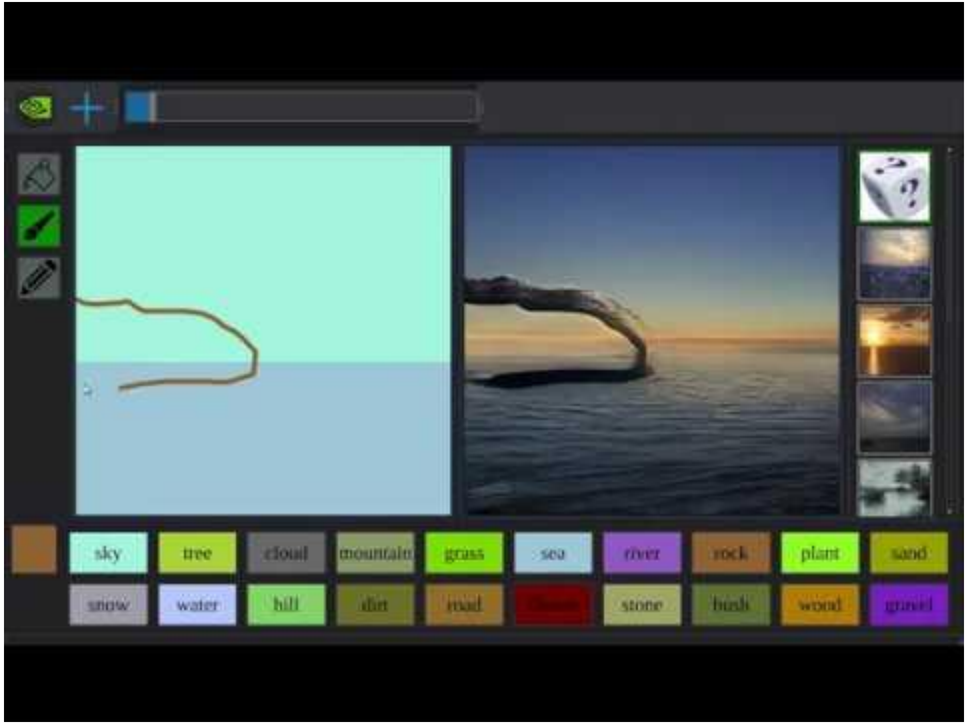
$$V_D(\hat{G}, D) = \underset{\boldsymbol{x}\sim q_{\mathrm{data}}(\boldsymbol{x})}{\mathrm{E}} [\min(0, -1 + D(\boldsymbol{x}))] + \underset{\boldsymbol{z}\sim p(\boldsymbol{z})}{\mathrm{E}} \left[\min\left(0, -1 - D\left(\hat{G}(\boldsymbol{z})\right)\right)\right]$$

$$V_G(G, \hat{D}) = \underbrace{\boxed{- \underset{\boldsymbol{z}\sim p(\boldsymbol{z})}{\mathrm{E}} \left[\hat{D}\left(G(\boldsymbol{z})\right)\right],}}_{\text{Loss of Generator}}$$

Image Encoder - Style Transfer



GauGAN - Interactive Demo



Datasets

COCO-stuff
Derived from COCO
# of classes: 182
# of training: 11.8k
# of validation: 5k

ADE20K
# of classes: 150
# of training: 20.2k
# of validation: 2k

ADE20K-outdoor
Only outdoor images

Cityscapes dataset
Street scenes in German cities
# of training: 3k
# of validation: 500

Flickr Landscapes
# of training: 41k
# of validation: 1k

Experimental Setup

Hyperparameters:

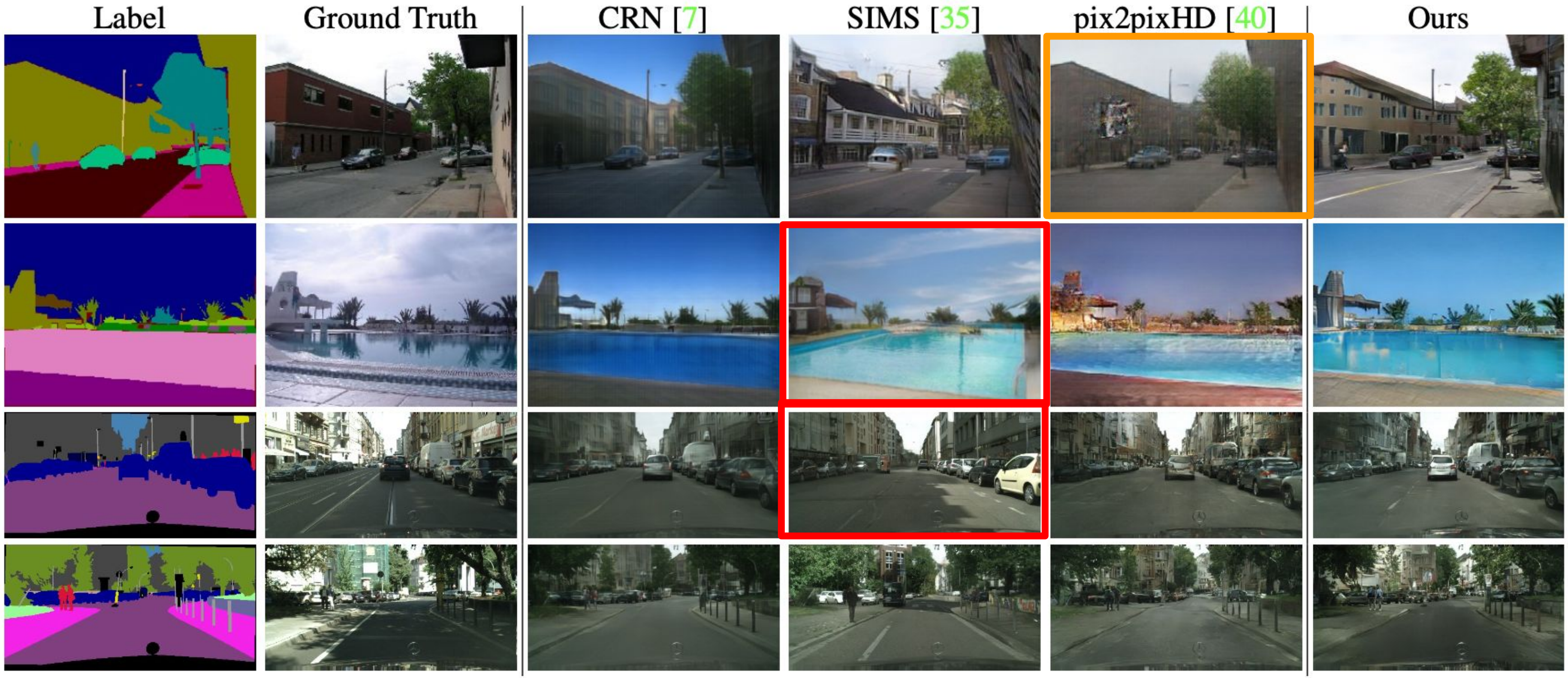
- Learning Rate (Generator & Discriminator): 0.0001 and 0.0004
- ADAM Optimizer: $\beta 1 = 0$, $\beta 2 = 0.999$

System Setup: NVIDIA DGX1 with 8 V100 GPUs

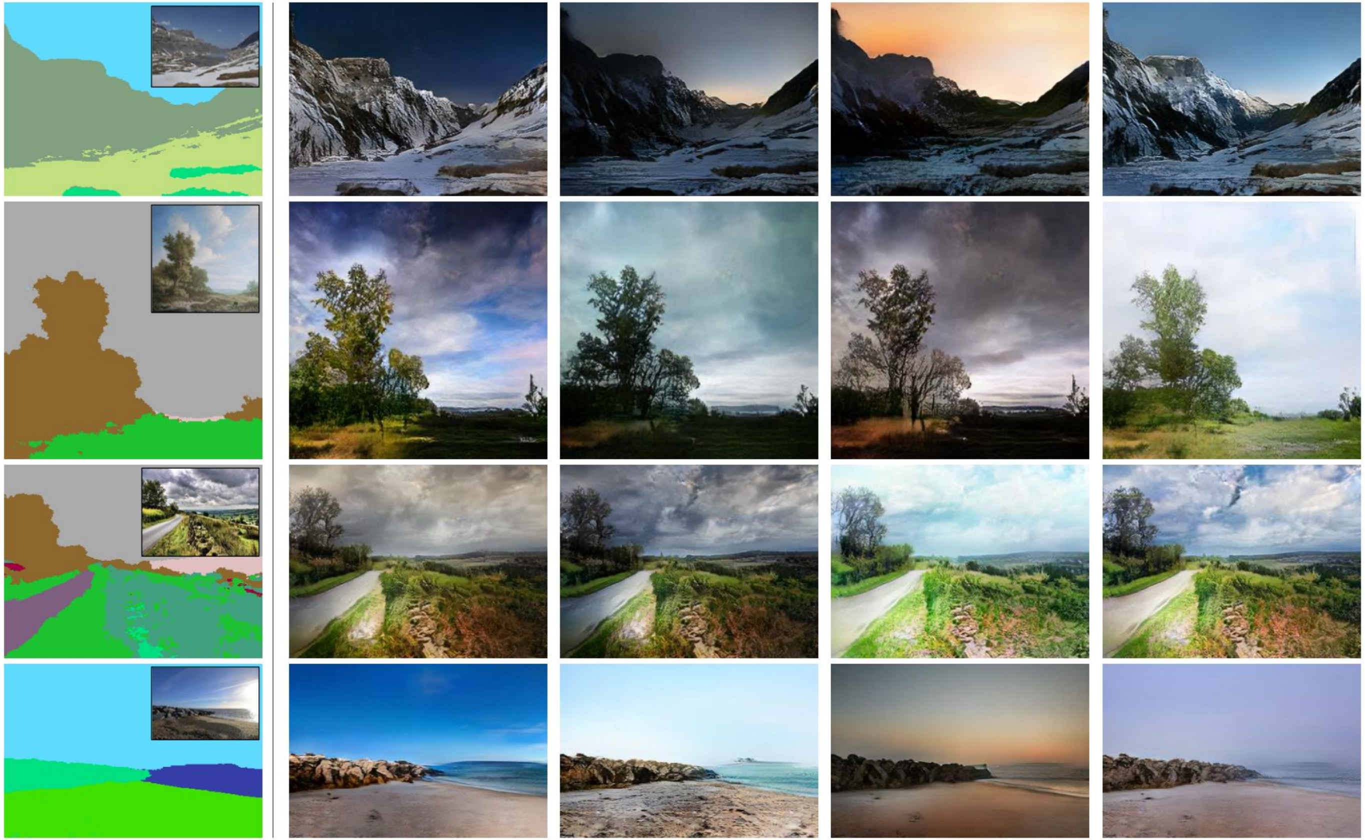
Baseline Models:

1. pix2pixHD: State-of-the-art GAN-based model
2. CRN (Cascaded Refinement Network): Deep Network refines the output from low to high resolution
3. SIMs (Semi-parametric IMage synthesis): Composites real segments from training set and refines boundaries

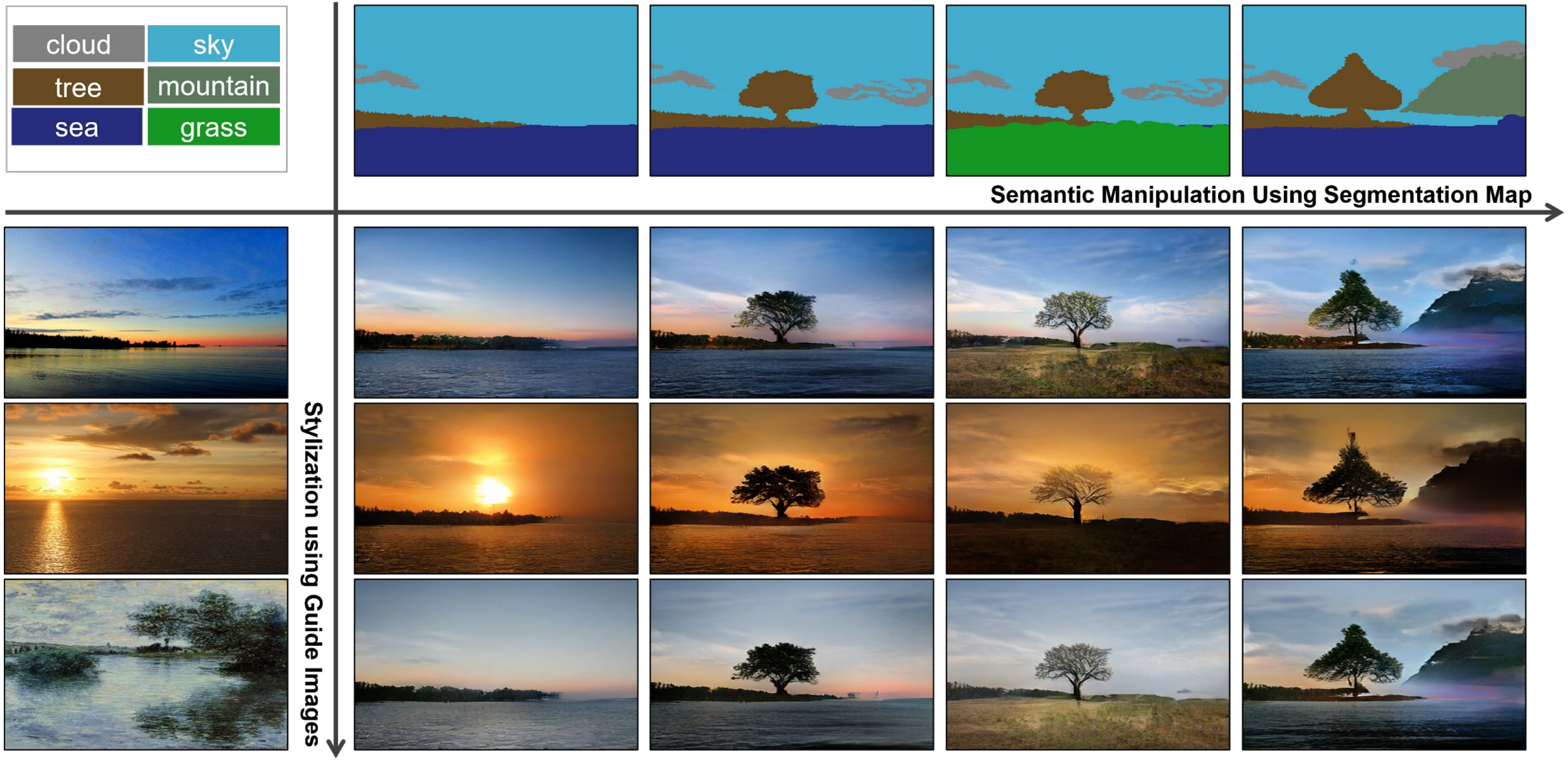
Results - Qualitative on ADE20K outdoor & Cityscapes



Results - Multimodal



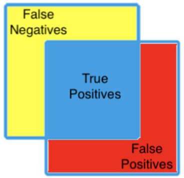
Results - Semantic and Style Control



Experiment - Metrics

Segment Label Comparison:

1. mIoU (mean Intersection over Union)
   IoU: Area of Overlap/Area of Union
   mIoU: Mean IoU for each class
2. Accu (Pixel Accuracy): % of pixels classified correctly i.e. TP



$$IoU = \frac{TP}{(TP + FP + FN)}$$

Images Comparison:

- FID (Fréchet Inception Distance) - Distance between the distributions of synthesized results and the distribution of real images.

For mIoU and pixel accuracy, higher is better. For FID, lower is better.

Result - Quantitative

Method	COCO-Stuff			ADE20K			ADE20K-outdoor			Cityscapes		
	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID
CRN [7]	23.7	40.4	70.4	22.4	68.8	73.3	16.5	68.6	99.0	52.4	77.1	104.7
SIMS [35]	N/A	N/A	N/A	N/A	N/A	N/A	13.1	74.7	67.7	47.2	75.5	**49.7**
pix2pixHD [40]	14.6	45.8	111.5	20.3	69.2	81.8	17.4	71.6	97.8	58.3	81.4	95.0
~~Ours~~ **Theirs**	**37.4**	**67.9**	**22.6**	**38.5**	**79.9**	**33.9**	**30.8**	**82.9**	**63.3**	**62.3**	**81.9**	71.8

Outperforms current leading methods in semantic segmentation scores

Result - Human Evaluation

Randomly generated 500 questions for each dataset, and each question answered by 5 different human evaluators

Dataset	Ours vs. CRN	Ours vs. pix2pixHD	Ours vs. SIMS
COCO-Stuff	79.76	86.64	N/A
ADE20K	76.66	83.74	N/A
ADE20K-outdoor	66.04	79.34	85.70
Cityscapes	63.60	53.64	51.52

Users preferred the results of the proposed method over the competing methods

Result - Ablation Study

- **pix2pixHD++:** Including all the techniques with pix2pixHD except SPADE
- **pix2pixHD++ w/ Concat:** Concat the segment mask input at all the intermediate layers
- **pix2pixHD++ w/ SPADE**: Strong baseline with SPADE
- Also, compare models with different capacity generators

Method	#param	COCO.	ADE.	City.
decoder w/ SPADE (Ours)	96M	**35.2**	38.5	62.3
compact decoder w/ SPADE	61M	**35.2**	38.0	**62.5**
decoder w/ Concat	79M	31.9	33.6	61.1
pix2pixHD++ w/ SPADE	237M	34.4	**39.0**	62.2
pix2pixHD++ w/ Concat	195M	32.9	38.9	57.1
pix2pixHD++	183M	32.7	38.3	58.8
compact pix2pixHD++	103M	31.6	37.3	57.6
pix2pixHD [40]	183M	14.6	20.3	58.3

Comparison done with respect to parameters used and mIOU scores

Result - SPADE generator variations

- Two inputs - Segmentation map, random noise input
- Varying kernel size, different capacity, and types of normalization

Method	COCO	ADE20K	Cityscapes
segmap input	35.2	38.5	62.3
random input	35.3	38.3	61.6
kernelsize 5x5	35.0	39.3	61.8
kernelsize 3x3	35.2	38.5	62.3
kernelsize 1x1	32.7	35.9	59.9
#params 141M	35.3	38.3	62.5
#params 96M	35.2	38.5	62.3
#params 61M	35.2	38.0	62.5
Sync Batch Norm	35.0	39.3	61.8
Batch Norm	33.7	37.9	61.8
Instance Norm	33.9	37.4	58.7

All scores are mIOU scores, Standard used bold ones

Contributions

- Semantics of image is captured by adding **SP**atially-**A**daptive **DE**normalization in the pix2pixHD architecture.

- Users can control both semantic and style for image synthesis

Strengths

- Diverse results for the same input, depending on the user’s interaction with objects (Style and Semantic support)
- SPADE resblk can be integrated with any existing architecture
- Reduced the number of trainable parameters and improved efficiency, when compared to previous state-of-the-art pix2pixHD
- Created semantic image synthesis interface(Live Demo) to interact and play on a canvas

Weaknesses

- Doesn’t train well for fine-grained details like faces
- Limited number of classes in demo
- Can be easily forced to generate un-natural shapes.

Future Work

- Can be extended to video synthesis
- Technique can be used for image super-pixelation task also
- In-the-wild technique can be integrated to put a constraint on the shapes of the object generated

Questions?