Course Presentation of CS 391R Robot Learning

Meta-Sim: Learning to Generate Synthetic Datasets

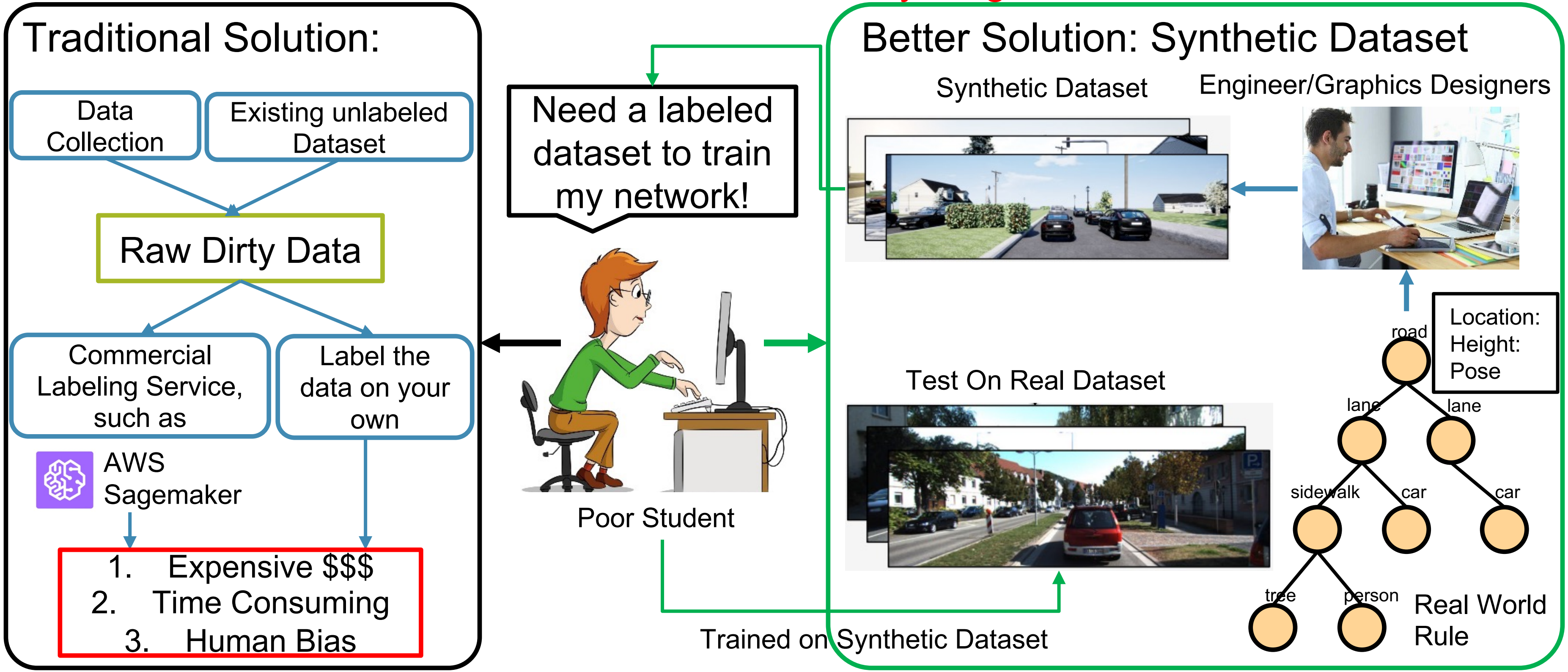
Authors: Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, Sanja Fidler

International Conference on Computer Vision (ICCV) 2019

Presenter: Changhan Ge
Department of Computer Science
The University of Texas at Austin
2021-10-05

Motivation

Anything more automatic and efficient?



Challenge: *Domain Gap*

Real World
Shinjuku Imperial Garden

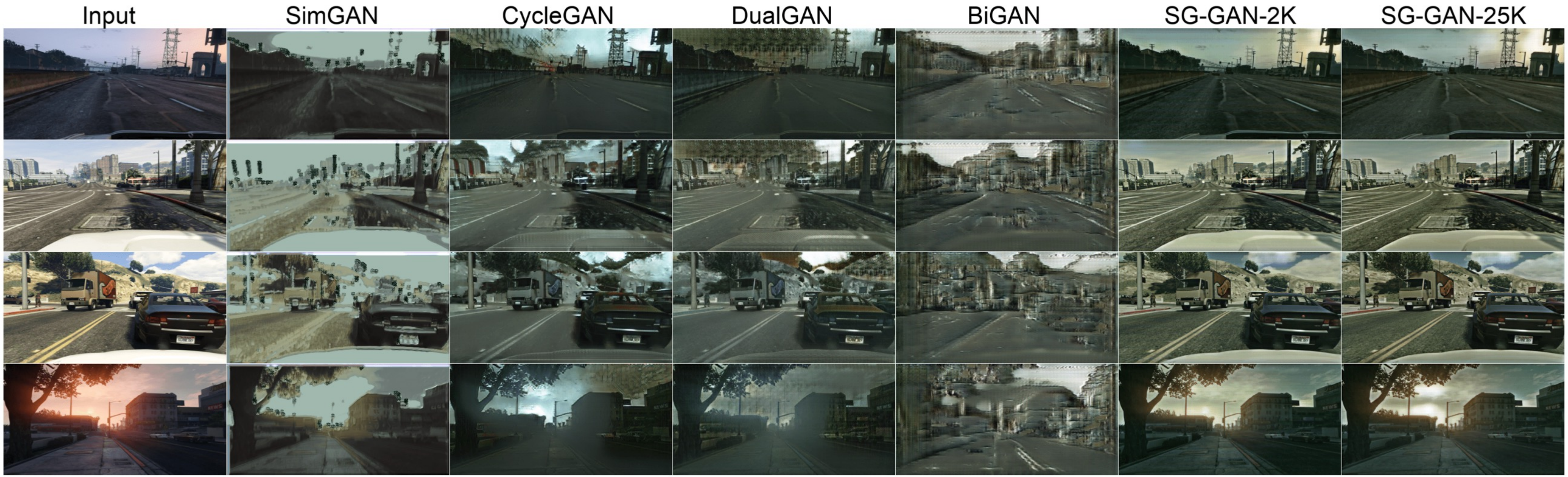
What we want [1]

What We Generated

[1] Makoto Shinkai et al. Garden of Words. 2013

Previous Efforts: Minimize the *Appearance Gap*

Key: Stylize the Synthetic Data



Domain Gap = Appearance Gap + ?

[2] Huang, Xun, et al. "Multimodal unsupervised image-to-image translation." *Proceedings of the European conference on computer vision (ECCV)*. 2018.
[3] Li, Peilun, et al. "Semantic-aware grad-gan for virtual-to-real urban scene adaption." arXiv preprint arXiv:1801.01726(2018).
[3] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.

Remained Challenge: *Content Gap*

What we have:
Traffic in Austin, TX

Key features:
1. Medium Traffic
2. Wider Road
3. Plain Terrain
4. Relative Low Building Density
5. Mostly Sunny
6. More pick-up trucks

What we want:

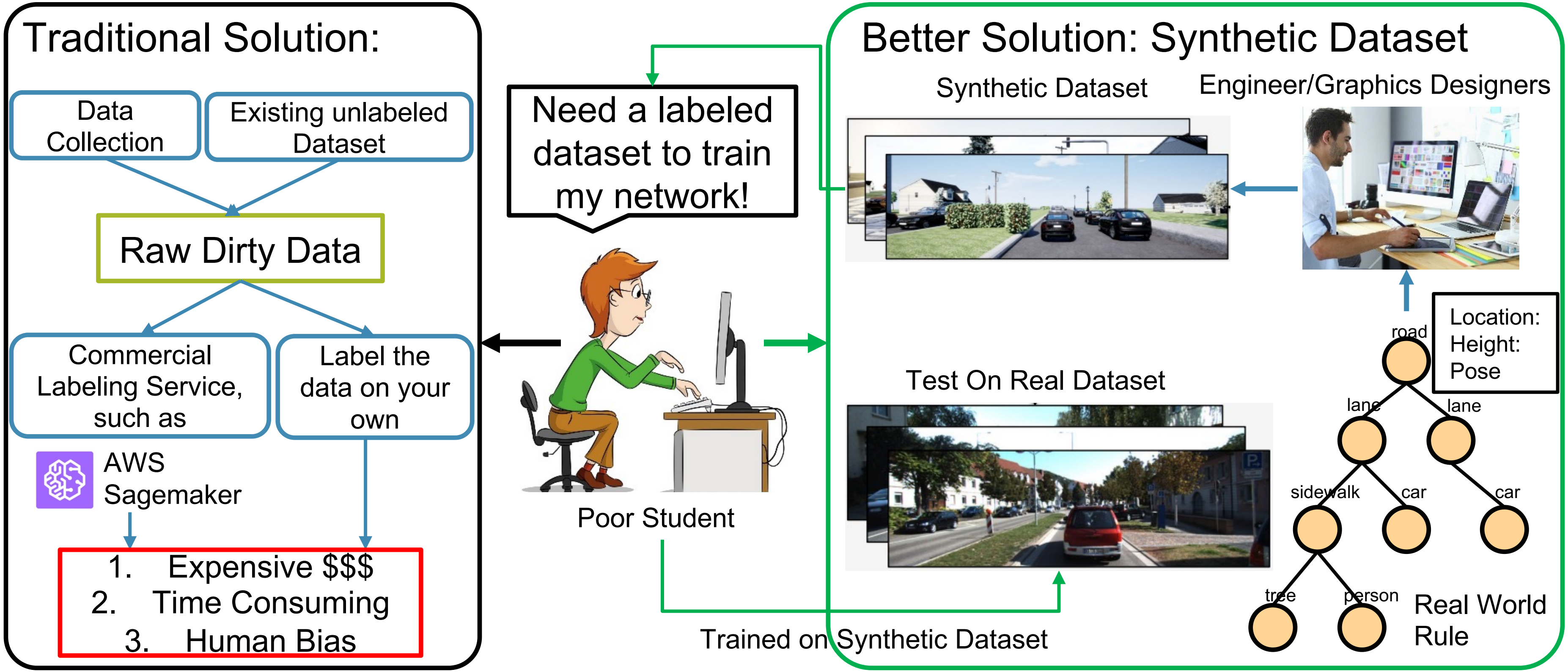
NYC Manhattan, NY
Heavy Traffic, Dense Building

San Francisco, CA
Heavy Traffic, Mountain City

Chongqing, China
Mountain City, Always Raining

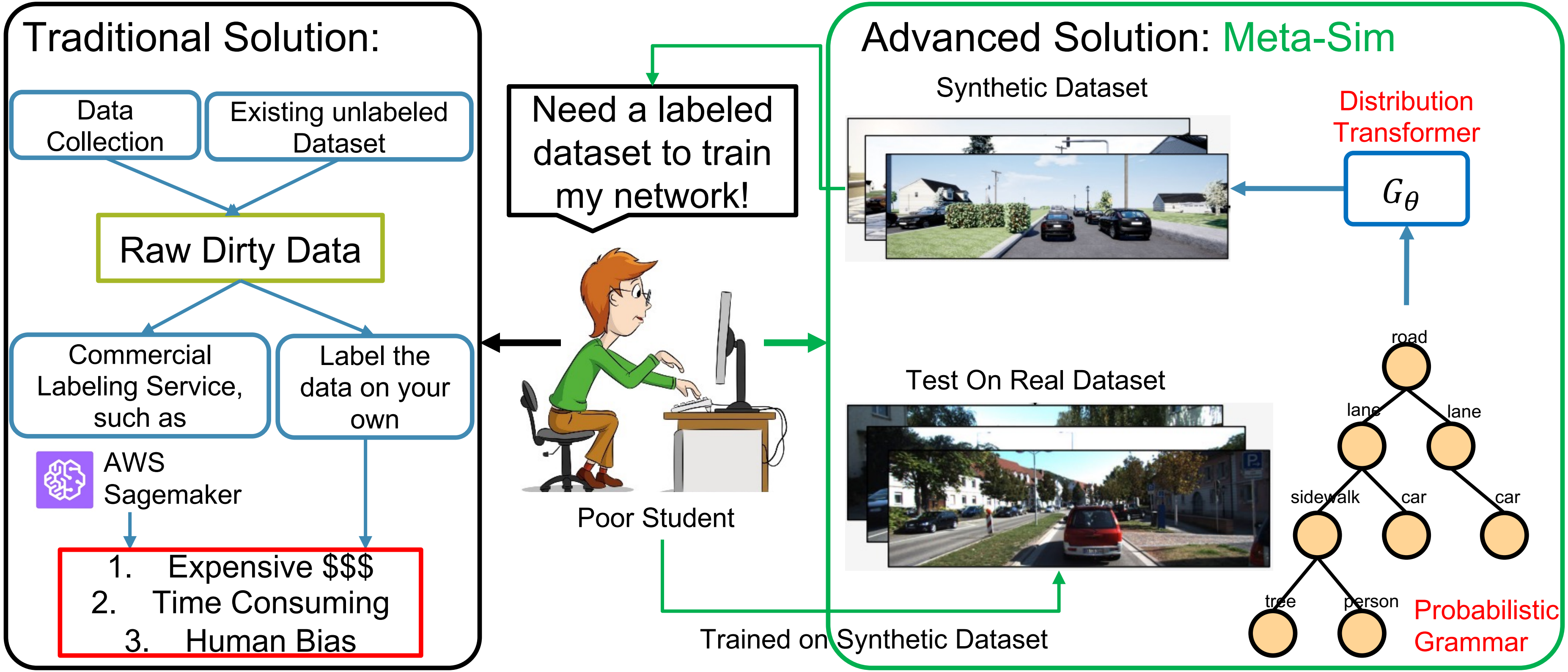
Rome, Italy
Compact Roads, Dense Building

Meta-Sim

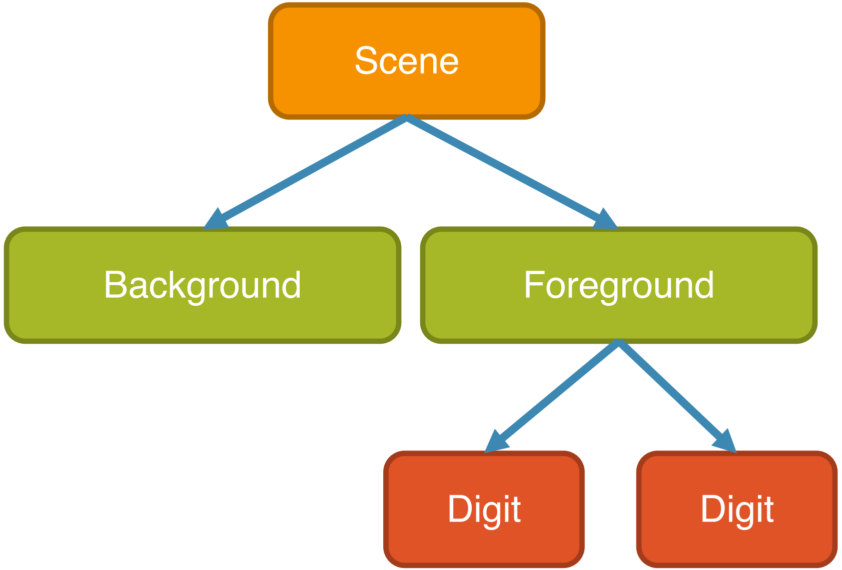


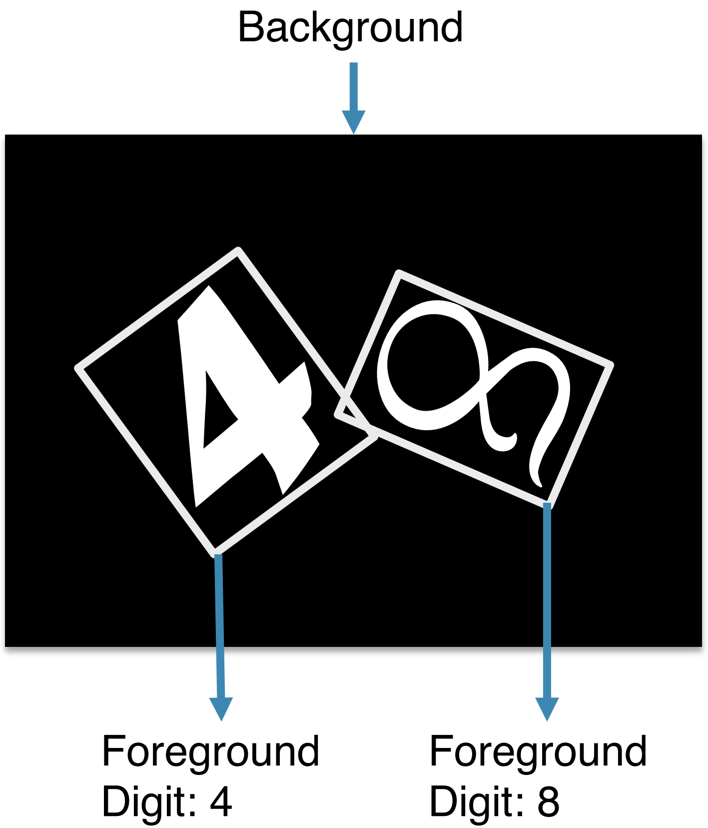
Traditional Solution:
Data Collection
Existing unlabeled Dataset
Raw Dirty Data
Commercial Labeling Service, such as
Label the data on your own
AWS Sagemaker
1. Expensive $$$
2. Time Consuming
3. Human Bias
Need a labeled dataset to train my network!
Poor Student
Trained on Synthetic Dataset
Better Solution: Synthetic Dataset
Synthetic Dataset
Engineer/Graphics Designers
Test On Real Dataset
Location:
Height:
Pose
road
lane
lane
sidewalk
car
car
tree
person
Real World Rule

Meta-Sim

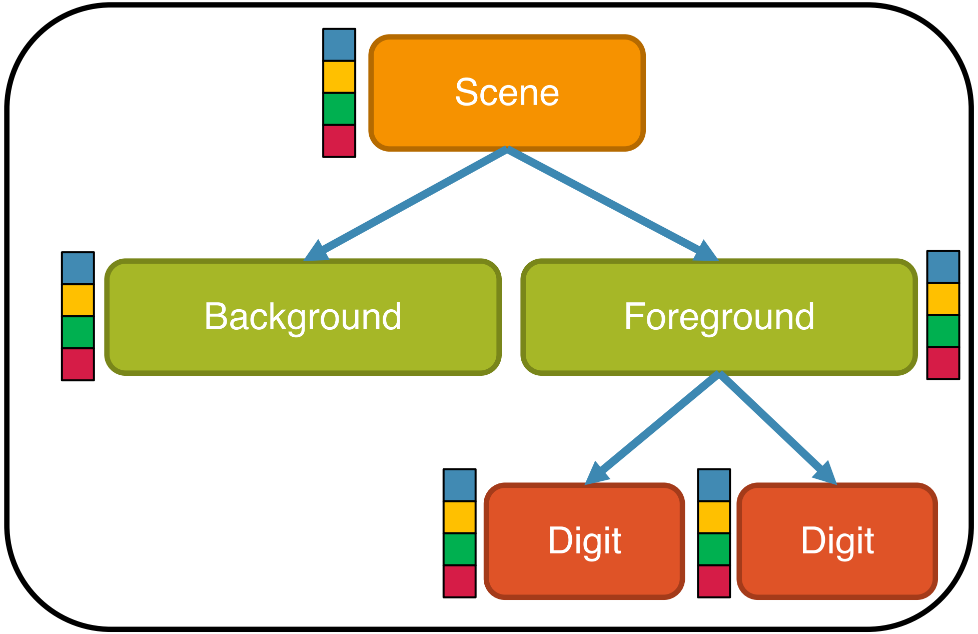

Traditional Solution:
Data Collection
Existing unlabeled Dataset
Raw Dirty Data
Commercial Labeling Service, such as
AWS Sagemaker
Label the data on your own
1. Expensive $$$
2. Time Consuming
3. Human Bias
Need a labeled dataset to train my network!
Poor Student
Trained on Synthetic Dataset
Advanced Solution: Meta-Sim
Synthetic Dataset
Distribution Transformer
G_θ
Test On Real Dataset
road
lane
lane
sidewalk
car
car
tree
person
Probabilistic Grammar

Anatomy of a Scene: *Structure*



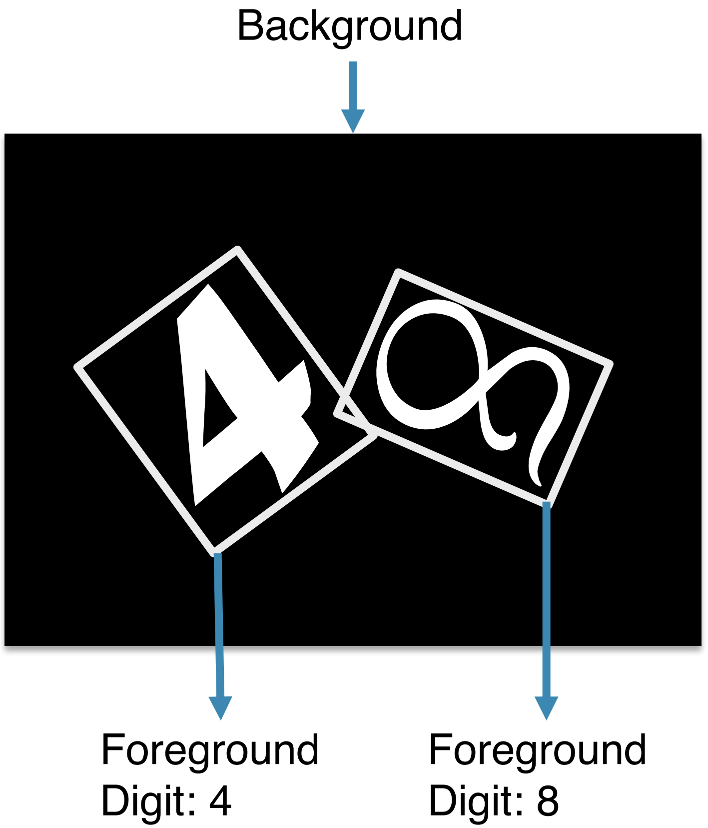


Anatomy of a Scene: *Structure*



Scene Graph

Class Rotation Location Size



Meta-Sim Modules:

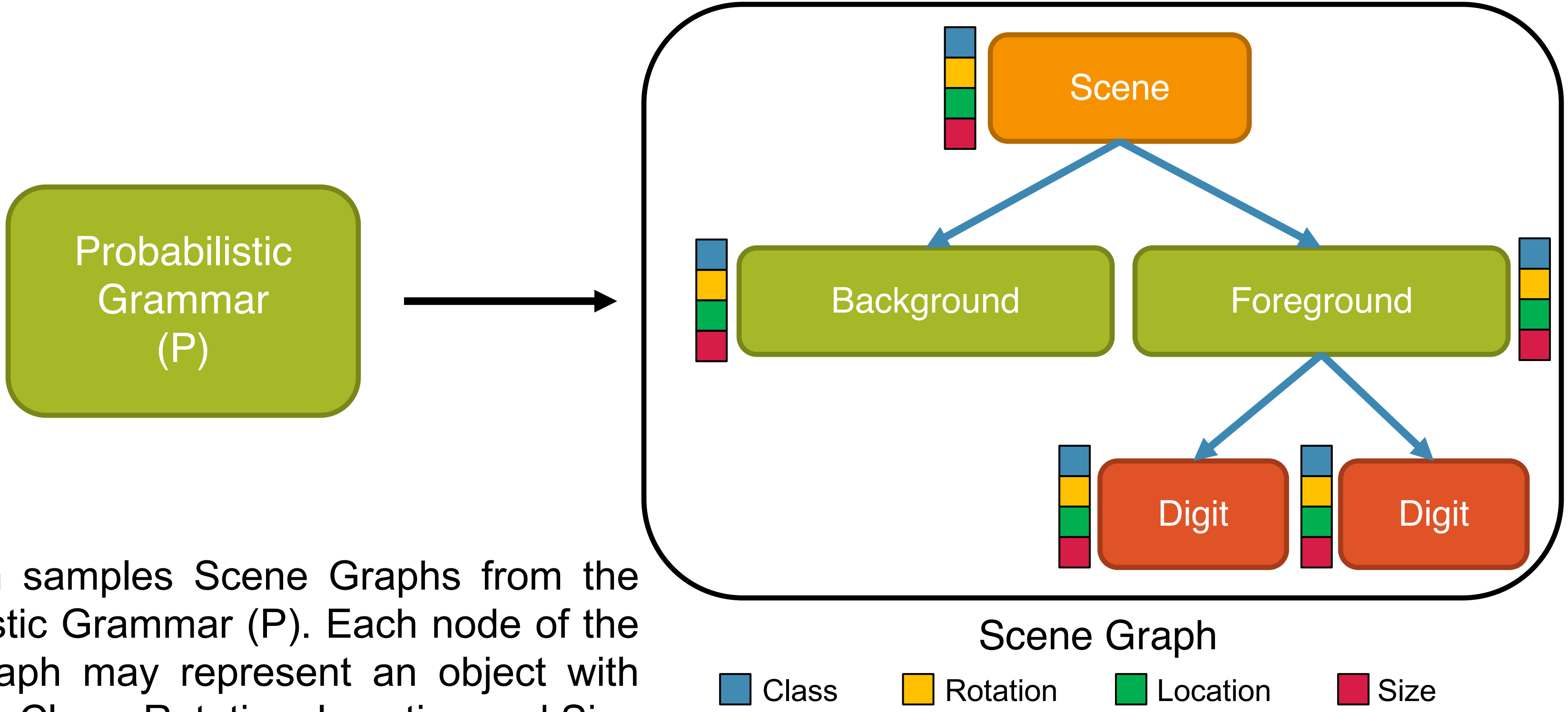
Probabilistic Grammar (P)

Renderer (R)

Distribution Transformer (DT)

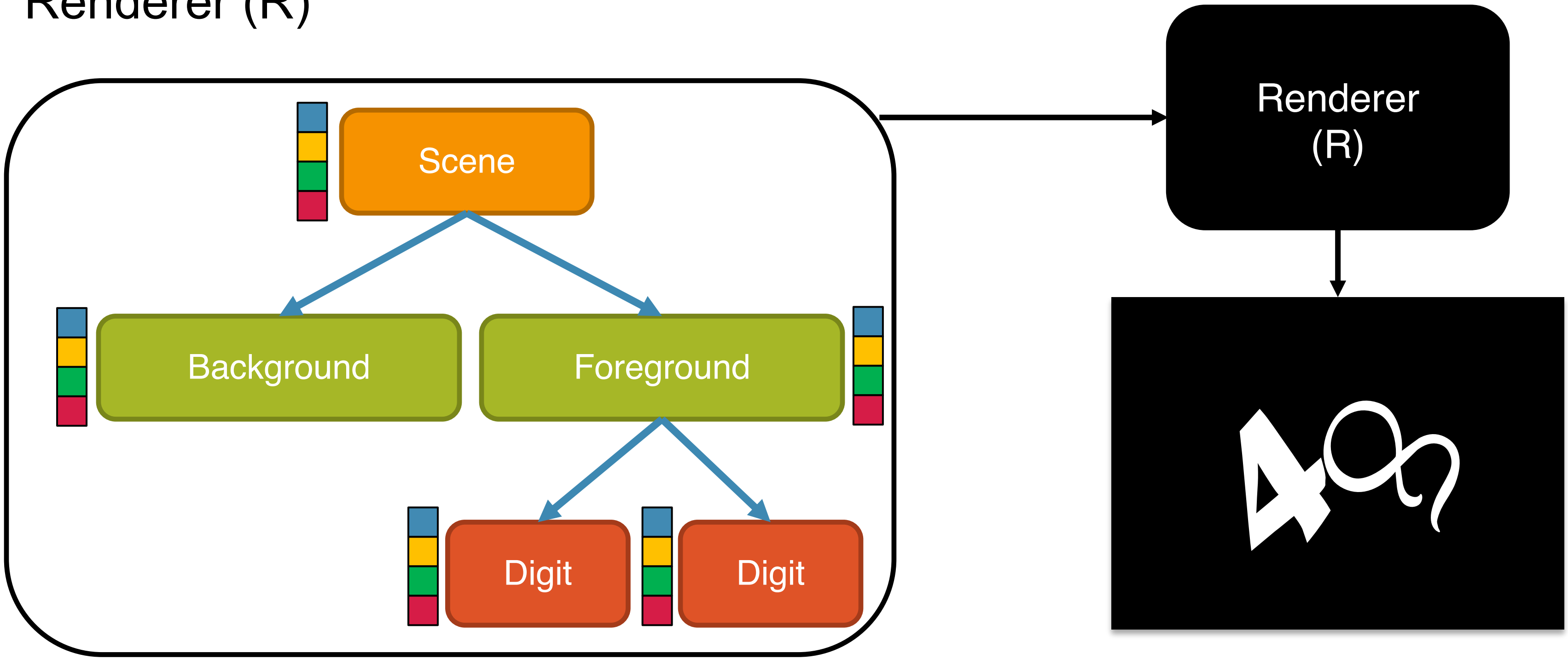
Task Network (TN)

Probabilistic Grammar (P)



Meta-Sim samples Scene Graphs from the Probabilistic Grammar (P). Each node of the scene graph may represent an object with attributes: Class, Rotation, Location and Size

Renderer (R)



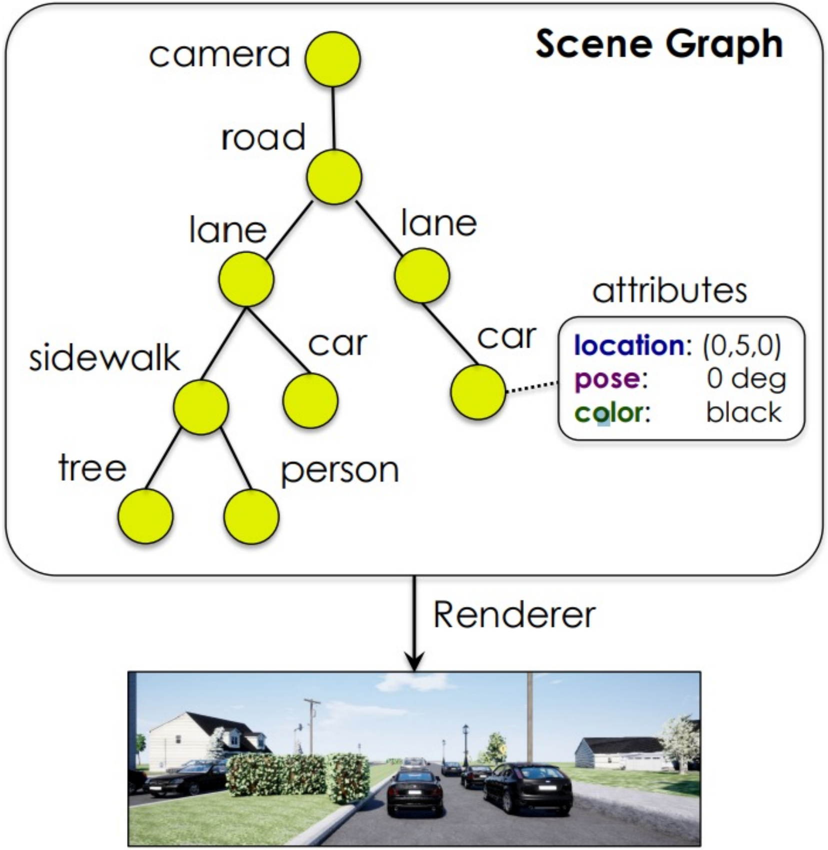
Scene Graph

Class Rotation Location Size

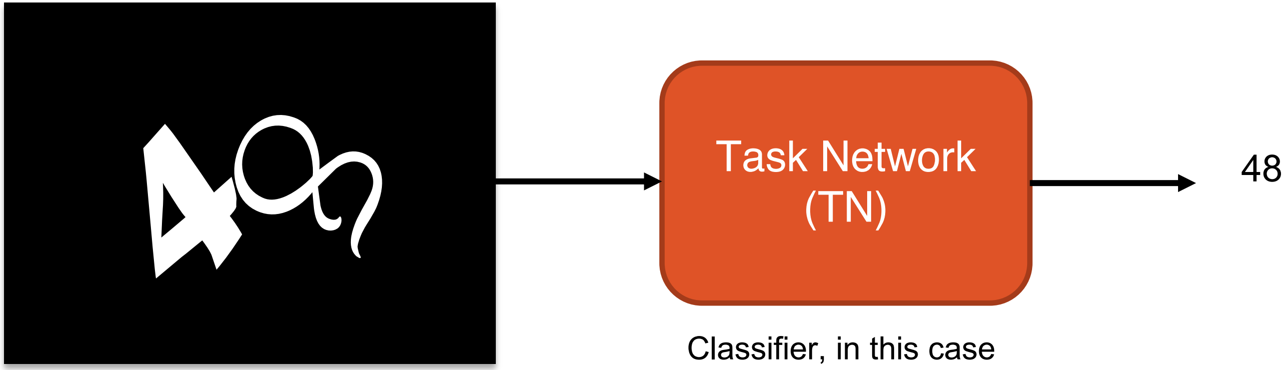
A non-differentiable renderer produces images corresponding to the sampled scene graph

Renderer (R)

A simple scene graph example
for a driving scene



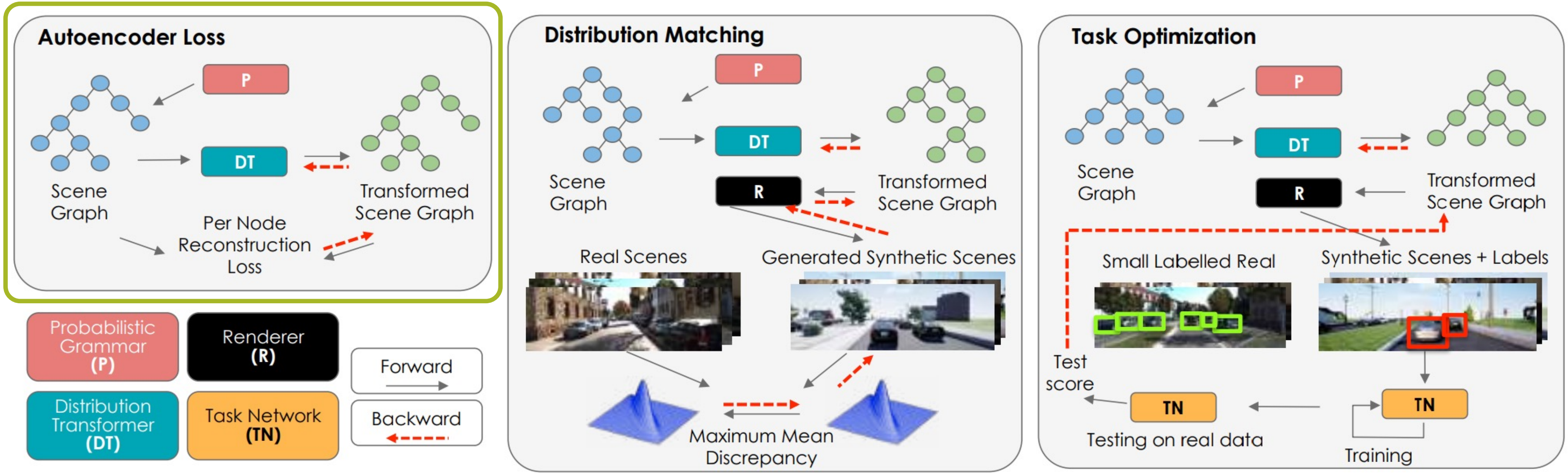
Task Network (TN)



Classifier, in this case

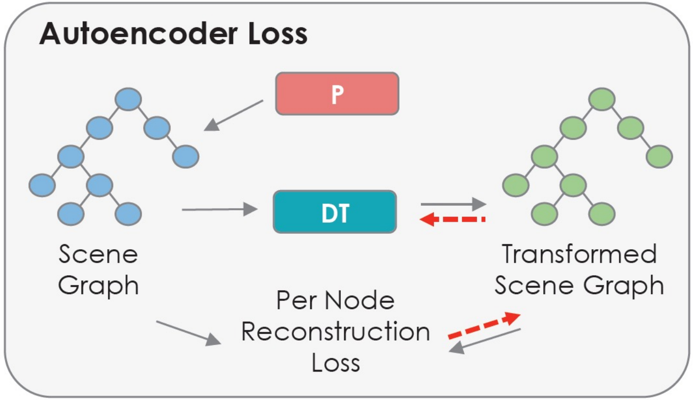
TN is a black box network learning a task on the synthetic dataset, for example classification here, and is used or tested against the real dataset

Training



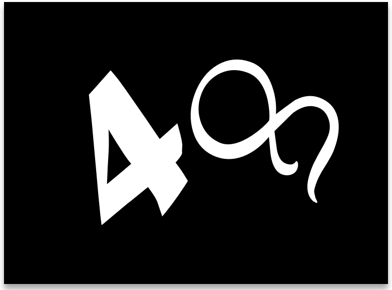
Autoencoder Loss
P
DT
Scene Graph
Transformed Scene Graph
Per Node Reconstruction Loss
Probabilistic Grammar (P)
Renderer (R)
Distribution Transformer (DT)
Task Network (TN)
Forward
Backward
Distribution Matching
P
DT
R
Scene Graph
Transformed Scene Graph
Real Scenes
Generated Synthetic Scenes
Maximum Mean Discrepancy
Task Optimization
P
DT
R
Scene Graph
Transformed Scene Graph
Small Labelled Real
Synthetic Scenes + Labels
Test score
TN
TN
Testing on real data
Training

Training – Pre-training: Autoencoder Loss



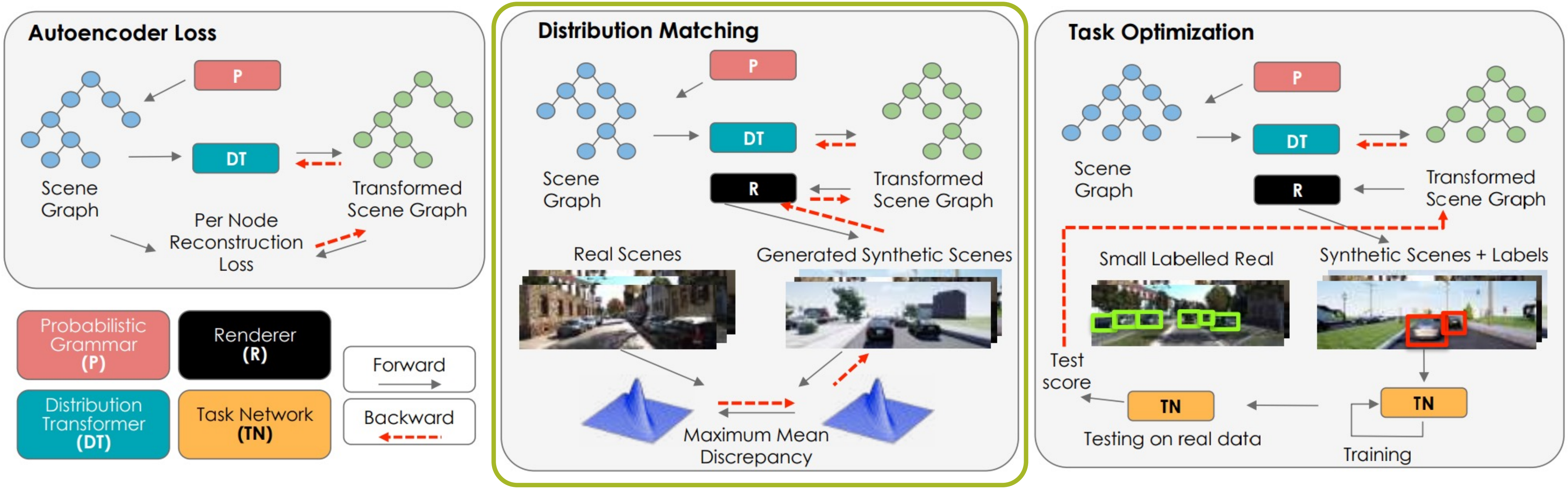
- Sample a scene graph from the Probabilistic Grammar
- Transform attributes of each node in the scene graph using the distribution transformer, which serves as an identity function as $G_\theta(s) = s$. Here, G_θ is a Graph convolutional network
- Compute the loss for attribute reconstruction per node (Cross Entropy for categorical attributes, L1 for continuous attributes)
- Backpropagate loss to DT

Example: MNIST dataset



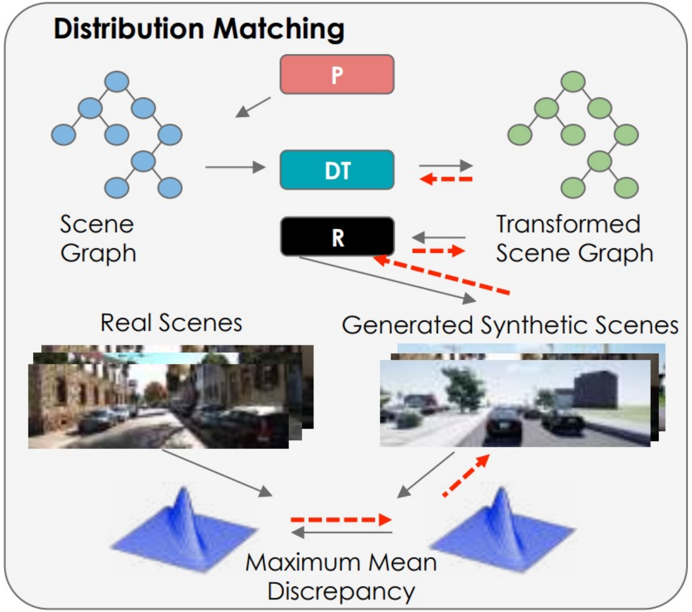
- Attribute set S_A = *[class, rotation, locationX, locationY, size].*
- *Class = [scene, background, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9]*
- *Rotation, locationX, locationY are float number between 0,1*
- $L_1 = \sum_{i=1}^{n} \left| y_{true} - y_{predict} \right|$
- $L_c = \sum_{i=1}^{n} y_{true} log y_{predict}$

Training



Autoencoder Loss
P
DT
Scene Graph
Transformed Scene Graph
Per Node Reconstruction Loss
Probabilistic Grammar (P)
Renderer (R)
Distribution Transformer (DT)
Task Network (TN)
Forward
Backward
Distribution Matching
P
DT
R
Scene Graph
Transformed Scene Graph
Real Scenes
Generated Synthetic Scenes
Maximum Mean Discrepancy
Task Optimization
P
DT
R
Scene Graph
Transformed Scene Graph
Small Labelled Real
Synthetic Scenes + Labels
Test score
TN
TN
Testing on real data
Training

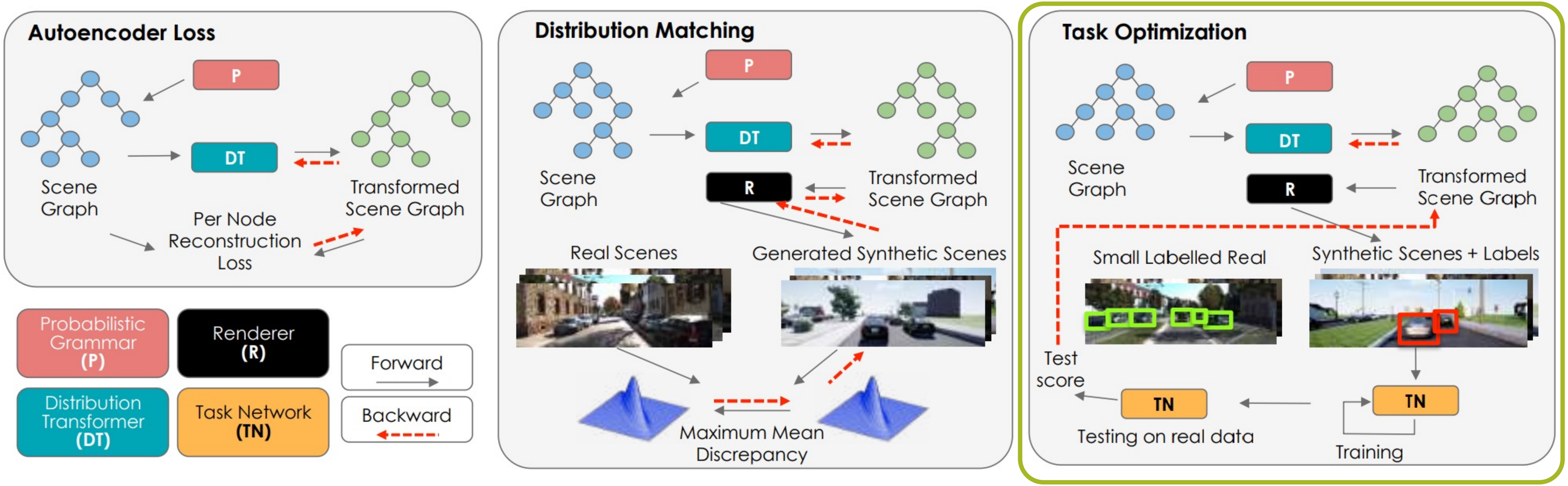
Training – Distribution Matching



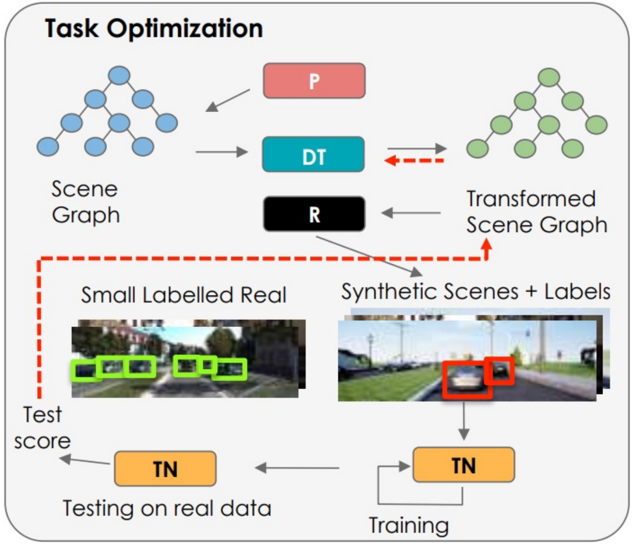
- Sample a scene graph and transform with the Distribution Transformer
- Render the transformed scene graph to get the synthetic images.
- Use a batch of generated images to compute maximum mean discrepancy (MMD) against a batch of images from the target dateset.
- Backpropagate the gradients to minimize MMD through the non-differentiable renderer R, by using finite differences (for each attribute in each node)

$$\mathcal{L}_{MMD^2} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{i'=1}^{N} k(\phi(X_\theta(s_i)), \phi(X_\theta(s_{i'}))) + \frac{1}{M^2}\sum_{j=1}^{M}\sum_{j'=1}^{M} k(\phi(X_R^j), \phi(X_R^{j'})) - \frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M} k(\phi(X_\theta(s_i)), \phi(X_R^j))$$

Training


Autoencoder Loss
P
DT
Scene Graph
Transformed Scene Graph
Per Node Reconstruction Loss
Probabilistic Grammar (P)
Renderer (R)
Forward
Distribution Transformer (DT)
Task Network (TN)
Backward
Distribution Matching
P
DT
R
Scene Graph
Transformed Scene Graph
Real Scenes
Generated Synthetic Scenes
Maximum Mean Discrepancy
Task Optimization
P
DT
R
Scene Graph
Transformed Scene Graph
Small Labelled Real
Synthetic Scenes + Labels
Test score
TN
TN
Testing on real data
Training

Training – Task Optimization



- Sample a scene graph, transform with the DT, and then render to get the corresponding synthetic scene.
- Use a set of generated scenes to train the task network for a few runs. Every epoch, the TaskNet is continue trained the from its last state instead of training from scratch.
- Test the trained TaskNet and get a performance score, for example, mean average precision for object detection
- Backpropagate the gradients directly, by using the REINFORCE score function estimator for the gradients.

Training – Pseudocode

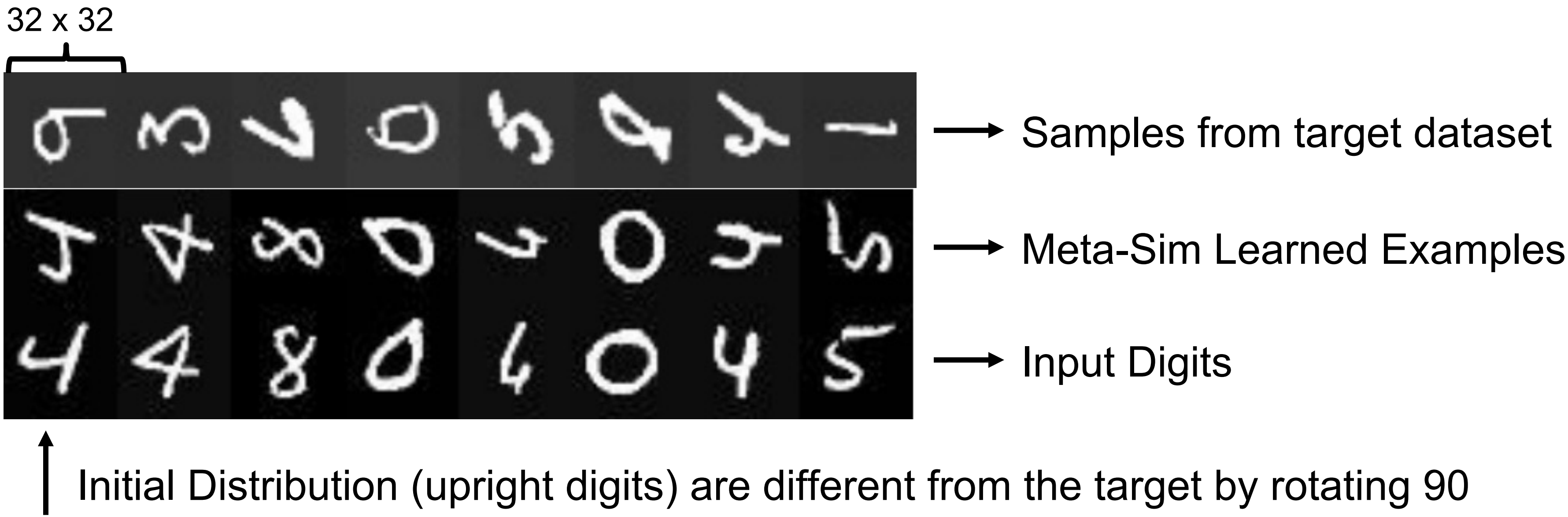
Algorithm: Meta-Sim's Meta Training Phase

Given: P, R, G_θ,	Probabilistic Grammar, Renderer, and GCN model
Given: TaskNet, X_R, V	Task Model, Real Images, Target Validation Data
Hyperparameters: E_m, I_m, B_m	Epochs, Iters, Batch Size
while $e_m \le E_m$ **do**	
loss = 0;	
data = []; *samples* = [];	Caching data & samples generated in epoch
while $i_m \le I_m$ **do**	
$S = G_\theta\left(sample(P, B_m)\right)$;	Generate B_m samples from P and transform them
$D = R(S)$;	Render images, labels from S
data += *D*; *samples* += *S*;	
$loss \mathrel{+}= L_{MMD2}(D, X_R)$;	MMD between generated and target real images.
end while	
TaskNet = train(TaskNet, *data*);	Train TaskNet on *data*
score = test(TaskNet, *V*);	Test TaskNet on target val
loss += -(score – moving_avg(score)) $logPG_\theta\,(samples)$	
$G_\theta = optimize(G_\theta, loss)$;	SGD step
end while	

Result - MNIST

Task: Digit Classification

Task Network: 2-layer CNN + 3 FC layer

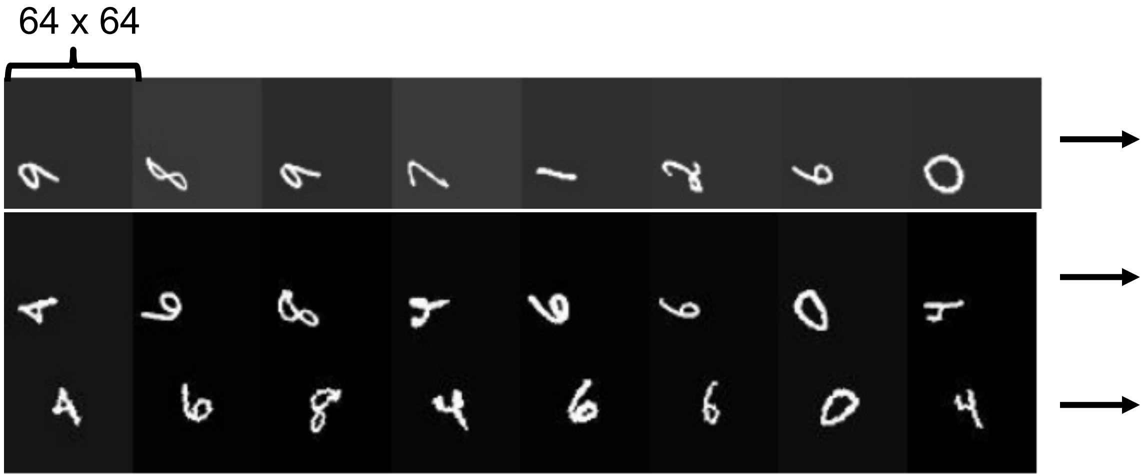


Initial Distribution (upright digits) are different from the target by rotating 90 degrees, which is to verity that Meta-Sim can learn the rotation correctly

Result – Rotated and Translated MNIST

Task: Digit Classification

Task Network: 2-layer CNN + 3 FC layer



Samples from target dataset (Rotate and Translate)

Meta-Sim Learned Examples

Input Digits

Result - MNIST

Task: Digit Classification

Task Network: 2-layer CNN + 3 FC layer

Classification Accuracy:

Data	Rotation	Rotation + Translation
Prob. Grammar	14.8	13.1
Meta-Sim	**99.5**	**99.3**

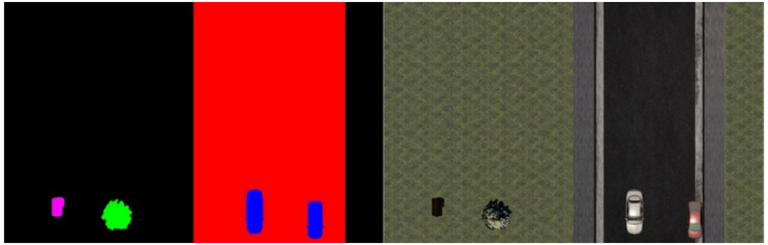
$> \frac{1}{10}$

The classifier trained on the dataset Meta-Sims generated performs significantly better than the the one trained on data directly output by probability grammar

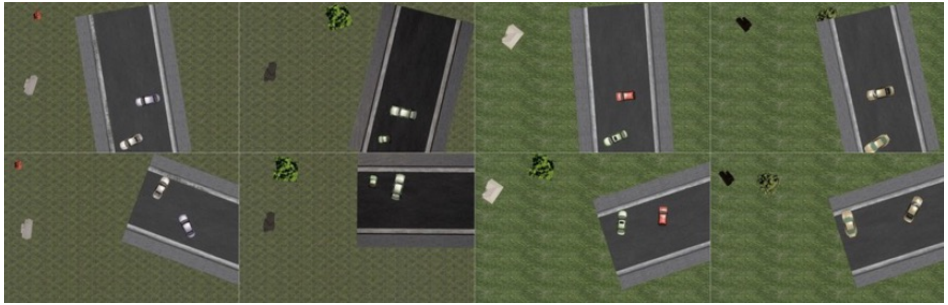
Meta-sim recovers the transformation causing the distribution gap

Result – Aerial2D

Task: Semantic Segmentation



Example label in Aerial 2D



Bottom: Input Scenes, Upper: Generated by Meta-Sim

Quantitative Result:

Data	Car	Road	House	Tree	Mean
Prob. Grammar	30.0	93.1	**98.3**	**99.7**	80.3
MetaSim	**86.7**	**99.6**	95.0	99.5	**95.2**

Table 2. Semantic segmentation results (IoU) on Aerial2D

Result – 3D driving

Task: Object Detection

Task Network: Mask-RCNN with Resnet 50FPN backbone (ImageNet Initialized)

Probabilistic Grammar | Meta-Sim Generated | KITTI Samples

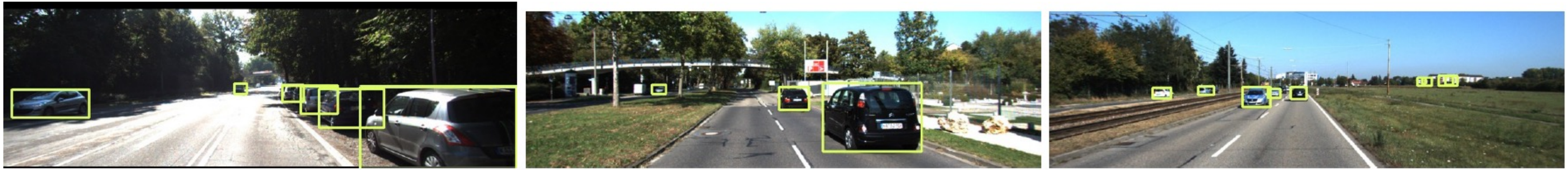
Result – 3D driving

Visualized Training Process

Result – 3D driving

Task: Object Detection

Task Network: Mask-RCNN with Resnet 50FPN backbone (ImageNet Initialized)



Car Detection result on the network trained with dataset generated by Meta-Sim

Car Detection result on the network trained with dataset generated by probabilistic Grammar

Result – 3D driving

Task: Object Detection

Task Network: Mask-RCNN with Resnet 50FPN backbone (ImageNet Initialized)

Quantitative Result:

Without Bridging the appearance gap

Data	Easy	Moderate	Hard
Prob. Grammar	63.7	63.7	62.2
MetaSim (Cars)	66.4	**66.5**	65.6
+ Camera	65.9	66.3	65.9
+ Context	65.9	66.3	66.0
+ Task Loss	**66.7**	66.3	**66.2**

With Bridging the appearance gap Using MUNIT

Data	Easy	Moderate	Hard
Prob. Grammar	71.1	**75.5**	65.3
Meta-Sim	**77.5**	75.1	**68.2**

AP @ 0.5 IOU for car detection on the KITTI validation dataset

Validates the Content Gap hypothesis

Result – 3D driving

Task: Object Detection

Task Network: Mask-RCNN with Resnet 50FPN backbone (ImageNet Initialized)

Quantitative Result:

Finetuning with 100 images from labeled KITTI dataset

TaskNet Initialization	Easy	Moderate	Hard
ImageNet	61.2	62.0	60.7
Prob. Grammar	71.3	72.7	72.7
Meta-Sim (Task Loss)	**72.4**	**73.9**	**73.9**

Table 5. Effect of finetuning on V

Pros and Cons

Pros:

- This works opens a new direction in computer vision community.
- Identifies the content gap.
- Delicate and Innovative design yields relatively good result.
- Wide and intensive evaluations

Cons:

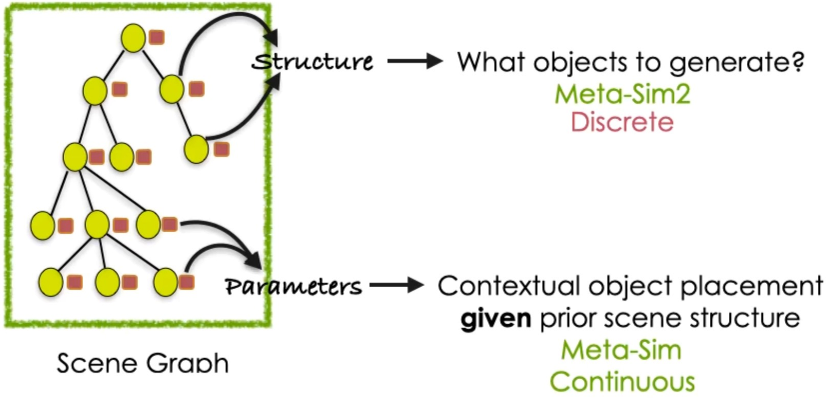
- Need Scalable Simulation
- Simulation need to adapt to the diversity of the real world
- Others:
  - This work is actually learning on the attributes, what about structure itself.
  - What about Differentiable Rendering?

Extended Readings: Upgraded version: Meta-Sim2

Devaranjan, Jeevan, Amlan Kar, and Sanja Fidler. "Meta-sim2: Unsupervised learning of scene structure for synthetic data generation." *European Conference on Computer Vision*. Springer, Cham, 2020.

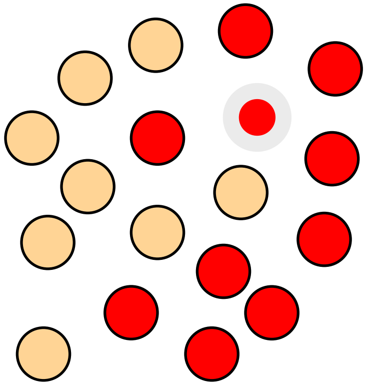
Major Difference:

- ❖ Directly learning from structure, instead of the attributes
- ❖ Unsupervised learning using a feature-based divergence between the target images and the render generated data.
- ❖ Used Reinforcement Learning to complete the task



Extended Readings: Upgraded version: Meta-Sim2

- In Meta-Sim, MMD is computed between the scenes, which provides a reward signal for a large batch of generated scenes. But no credit assignment



- In Meta-Sim2
  - Step 1: Compute the likelihood of under all = p (synthetic)
  - Step 2: Compute the likelihood of under all = p (real)
  - Step 3: Use log (likelihood – ratio) as reward per scene

Approximation: Compute Likelihood non parametrically using Kernel Density Estimation (RBF Kernel)

Approximation: Compute Likelihood using a batch of generated real and synthetic samples

Scenes are point clouds in a feature space

Real Scenes

Synthetic Scenes

Extended Readings: Upgraded version: Meta-Sim2

Qualitative Result:

Method	Structure	Parameters	Easy	Medium	Hard	KID [5]	FID [27]
Prob. Grammar	Prior	Prior	63.7	63.7	62.2	0.066	106.6
Meta-Sim* [30]	Prior	Learnt	66.5	66.3	65.8	0.072	111.6
Ours	Learnt	Learnt	**67.0**	**67.0**	**66.2**	**0.054**	**99.7**

Summary

- Problem: Meta-Sim is a model and training strategy to generate labeled synthetic visual dataset.
- Significance: AI researches require large amount of labeled dataset.
- Hardship: The domain gap between the generated synthetic image and the real image are huge.
- Key Insights:
  - Content gap hypothesis.
  - Replace the real-world rule and human efforts with probabilistic grammar and a distribution transformer.
  - Find an effective way to represent the scenes using scene graph.
  - Build a delicate model which combines many different components to achieve the goal.
- Demonstrations: Existence of content gap. The model yields the state-of-the-art performance since it is the first work in this topic.

Thanks! Q&A