Learning Canonical Representations for Scene Graph to Image Generation


Roei Herzig
TAU


Amir Bar
TAU


Huijuan Xu
Berkeley

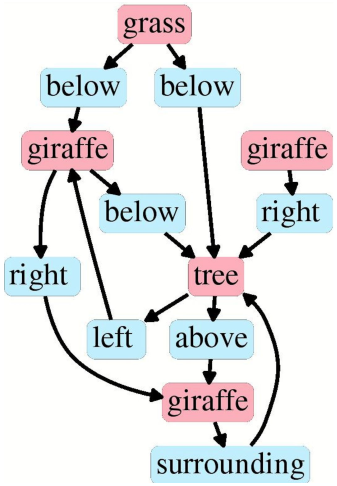

Gal Chechik
Bar-Ilan/Nvidia


Trevor Darrell
Berkeley


Amir Globerson
TAU

Image Generation from Scene Graphs (SGs)

Goal: learn a function **F** for SG-to-image



F

Input - scene graph

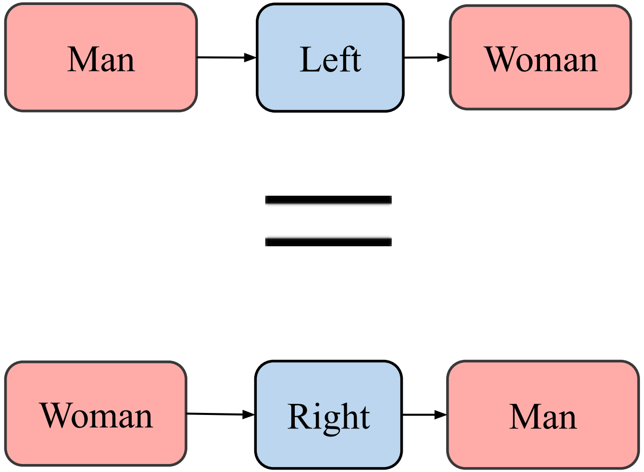
Output - generated image

Task proposed in: Image Generation from Scene Graphs, Johnson et al., CVPR 2018.

Limitation of Current Models

Equivalent inputs lead to different predictions

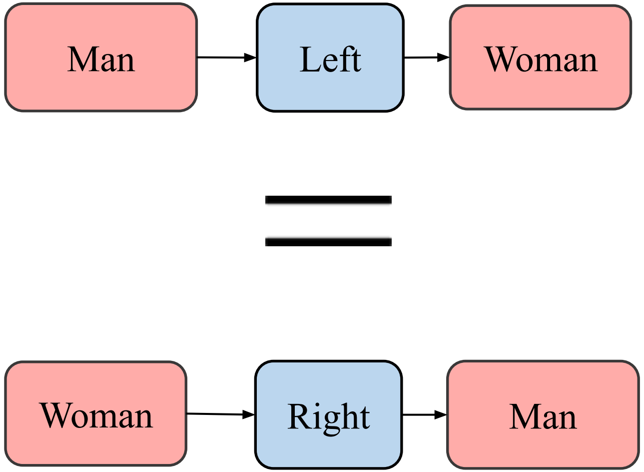
Semantically equivalent



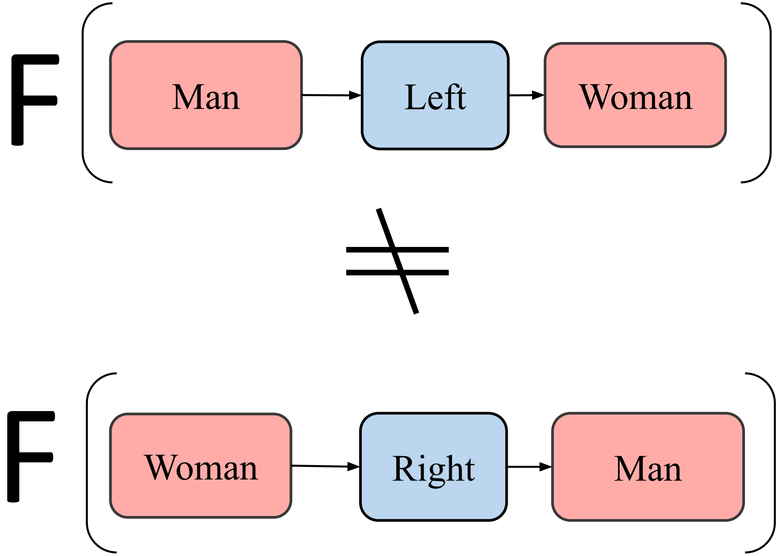
Limitation of Current Models

Equivalent inputs lead to different predictions

Semantically equivalent



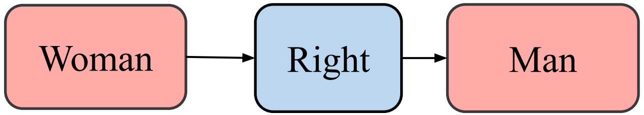
Two **different** images



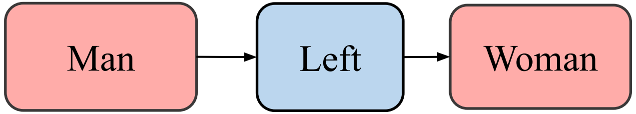
Contributions

Our model learns a **canonical** graph representation from the data that **obtains stronger invariance** properties

Input SG #1

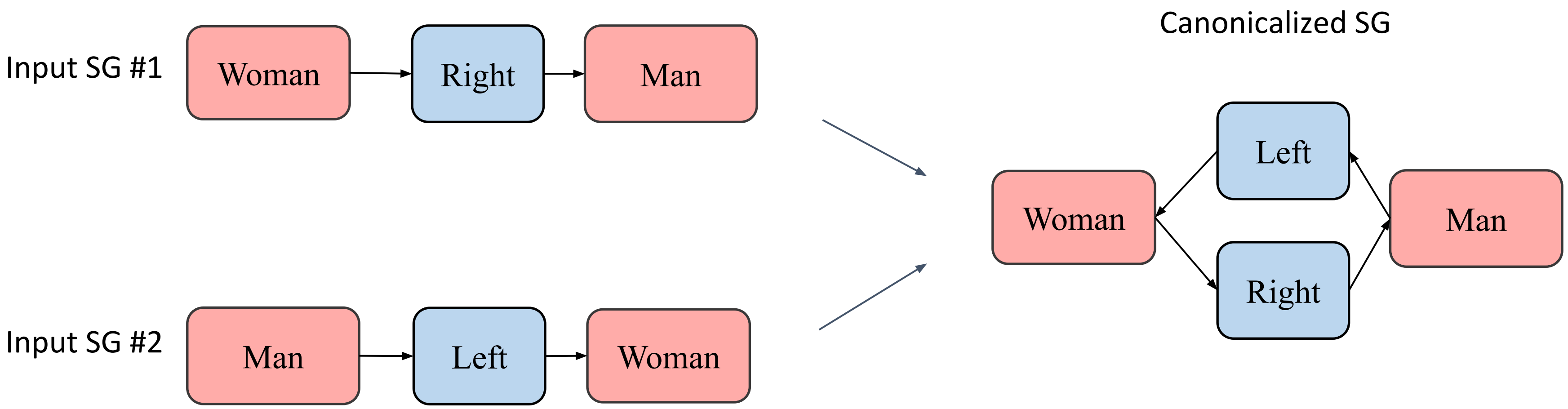


Input SG #2



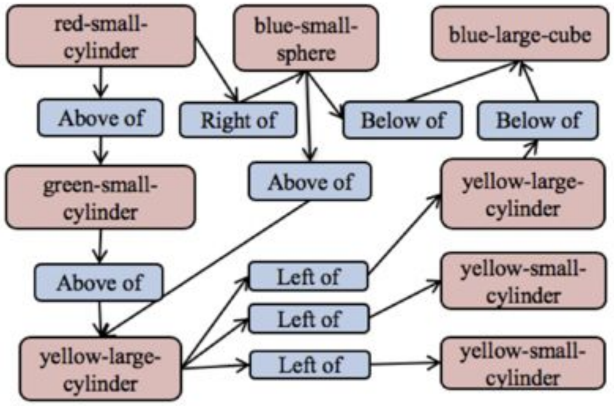
Contributions

Our model learns a **canonical** graph representation from the data that **obtains stronger invariance** properties



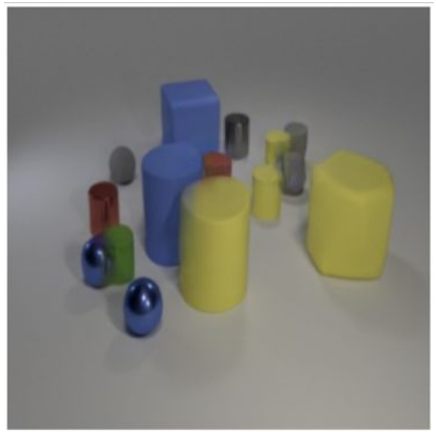
Contributions

This leads to improved performance on **large SGs**, **robustness to noise** in the input SG, and better **generalization**.



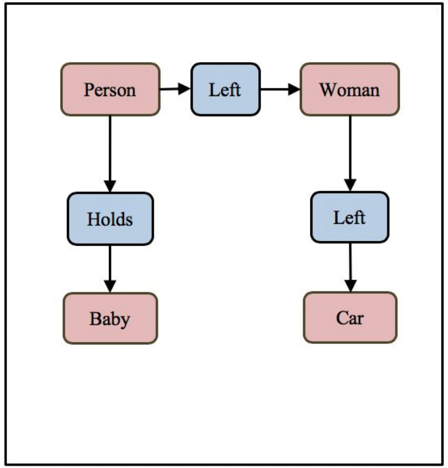
Input Scene Graph (partial)

Predicted Image (sg2im)

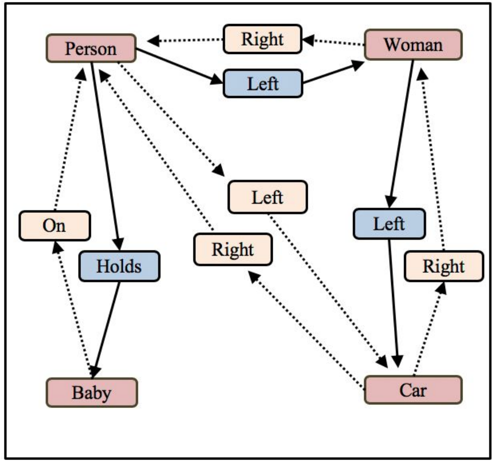


Predicted Image (ours)

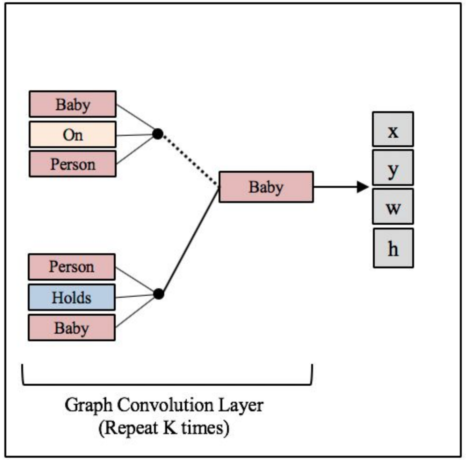
High Level Architecture



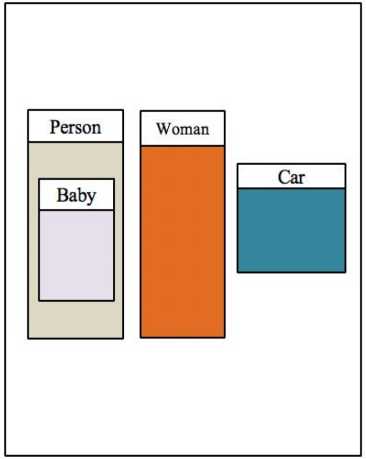
Input Scene Graph



Scene Graph Canonicalization



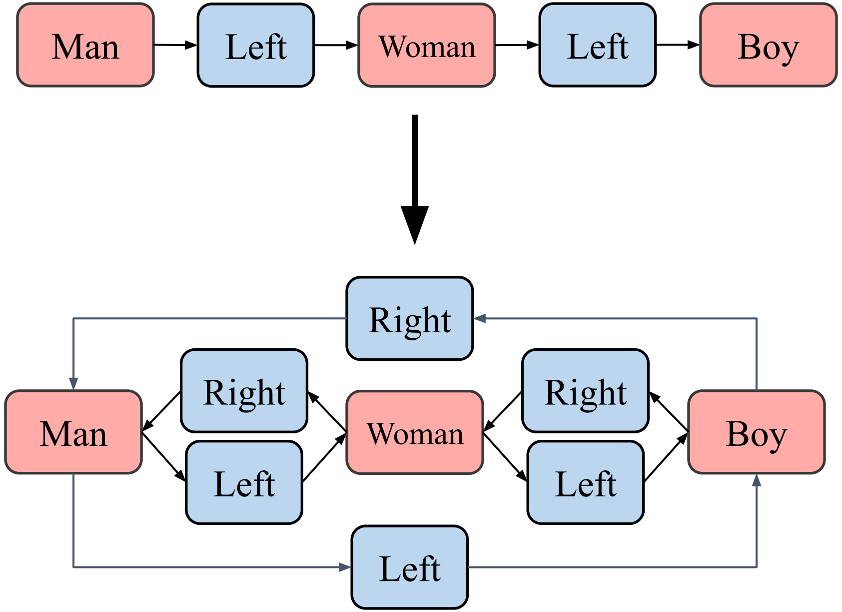
GCN



Scene Layout

Scene Graph Canonicalization

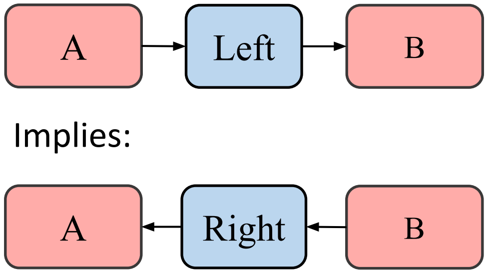
What should be the canonical form of an input scene graph?



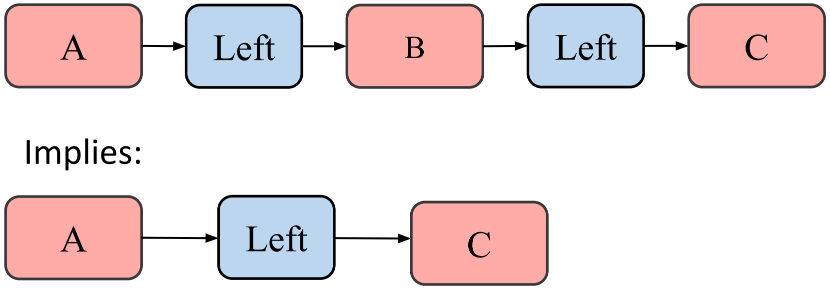
A natural choice is the "relation-closure", the graph containing all the possible implied edges.

Scene Graph Canonicalization

We deal with two types of completion rules. Converse completion and transitive completions.



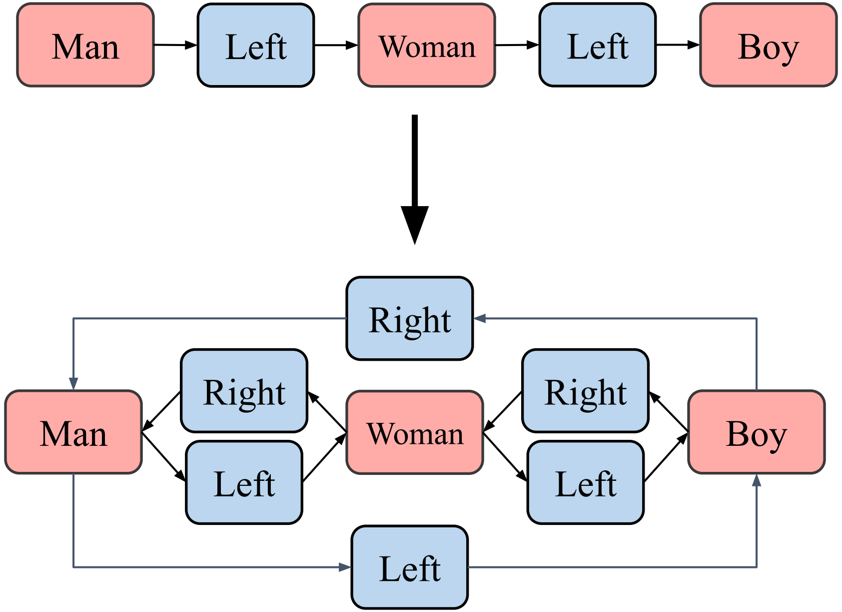
Converse Relations



Transitive Relations

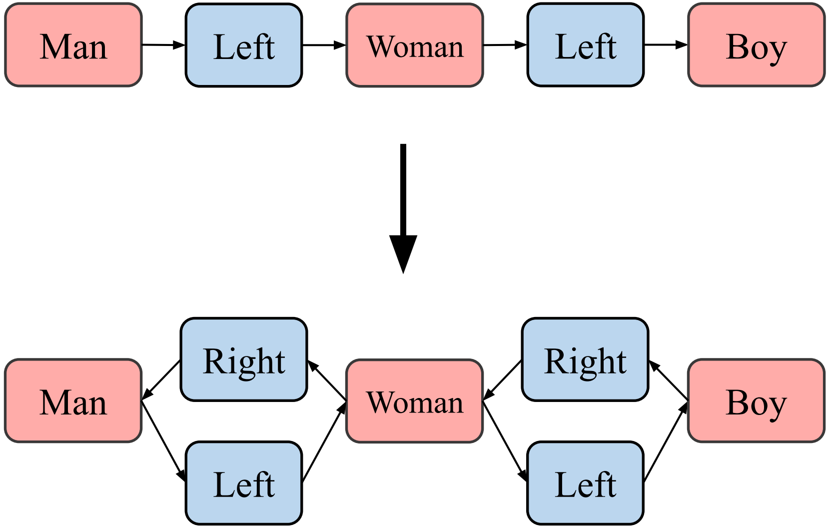
Scene Graph Canonicalization

Goal: given an input scene graph, compute its *relation-closure*



"relation-closure"

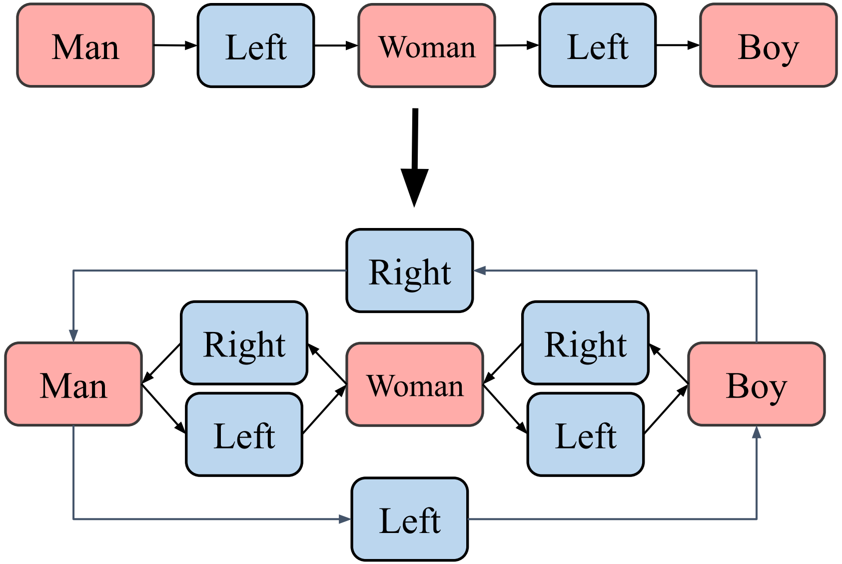
Scene Graph Canonicalization



Step 1: converse completions

For every edge, we complete its converse edges.

Scene Graph Canonicalization

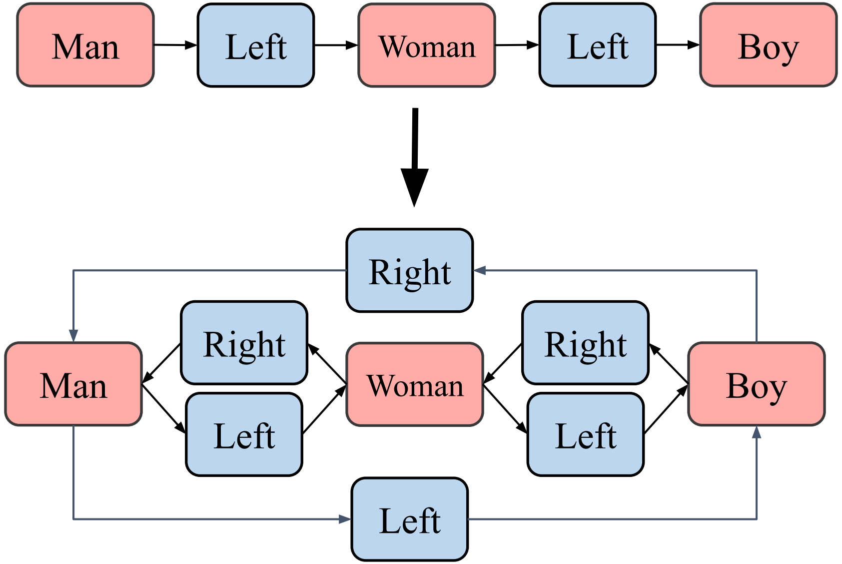


Step 1: converse completions

Step 2: transitive completions

For every transitive relation, we construct missing transitive edges

Scene Graph Canonicalization



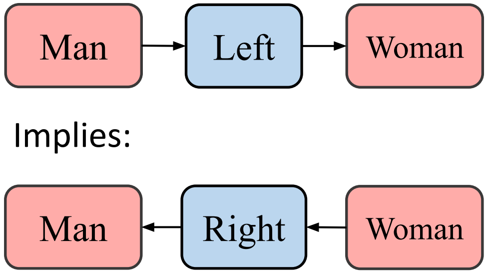
Step 1: converse completions
Step 2: transitive completions

This procedure produces the *relations-closure*.

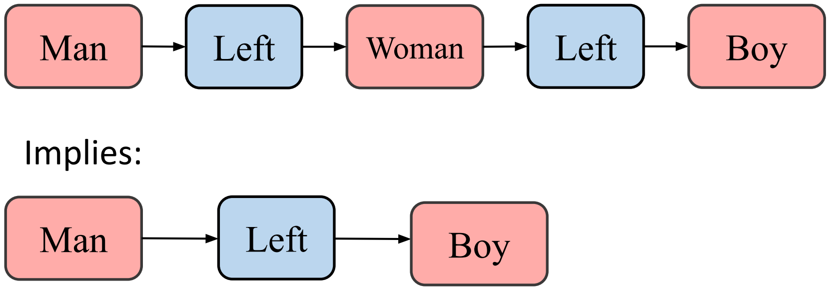
Scene Graph Canonicalization

An assumption of the SGC is that **converse and transitive completion rules are known**.

We next show how to learn this from data.



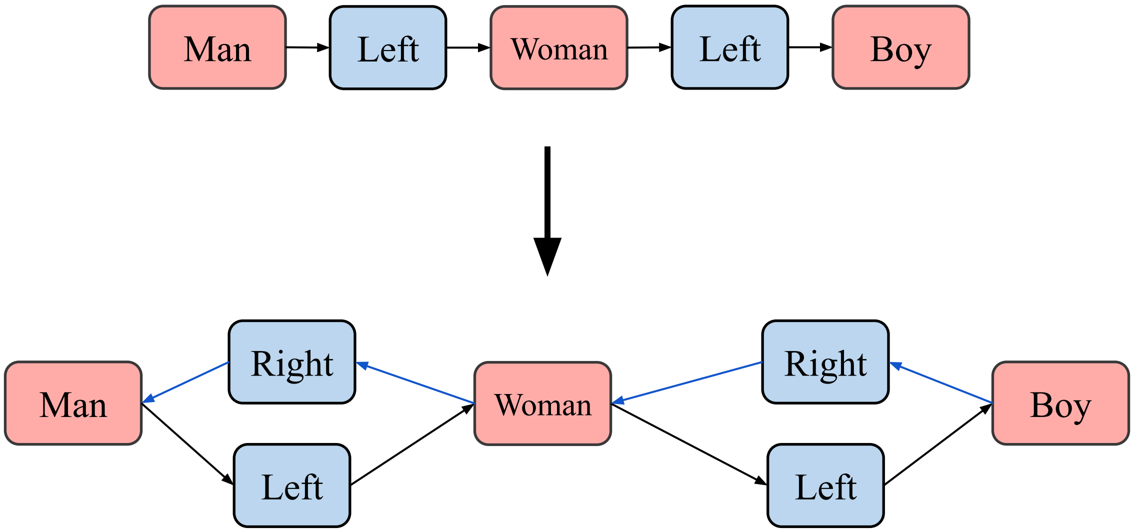
Converse Rules



Transitive Rules

Weighted Scene Graph Canonicalization

Step 1: converse completions



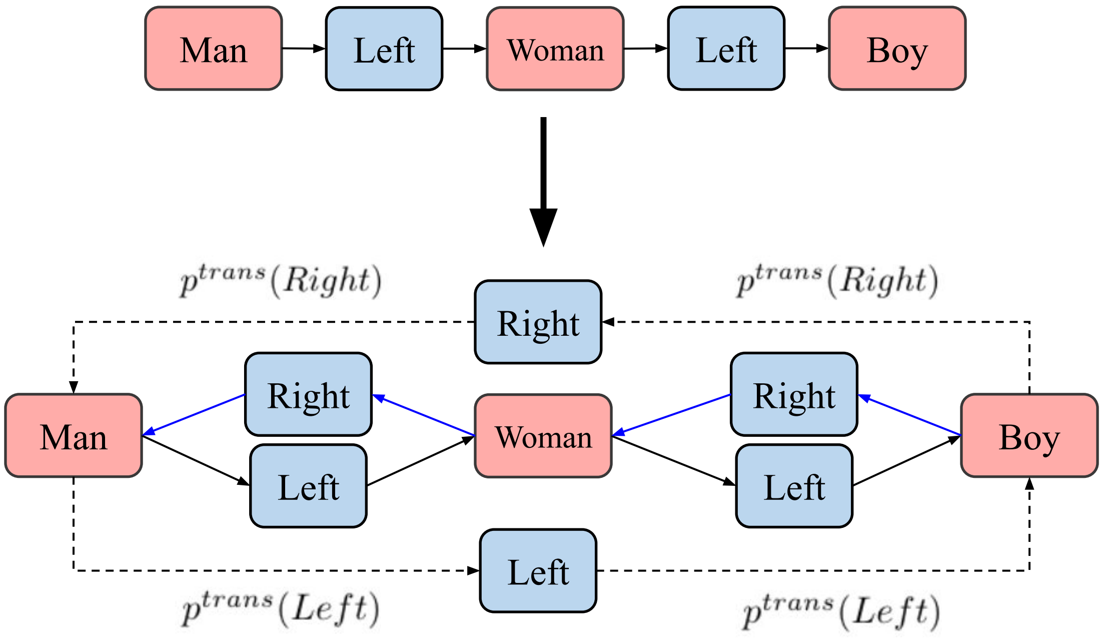
For every relation pair *r* we can learn the probability that *r'* is converse to it.

$$p^{conv}(r'|r) = \frac{e^{\theta^{conv}_{r,r'}}}{\sum_{\hat{r}\in\mathcal{R}\cup\phi} e^{\theta^{conv}_{r,\hat{r}}}}$$

To complete edges, for every edge, we sample from its corresponding converse distribution.

Weighted Scene Graph Canonicalization

Step 2: transitive completions

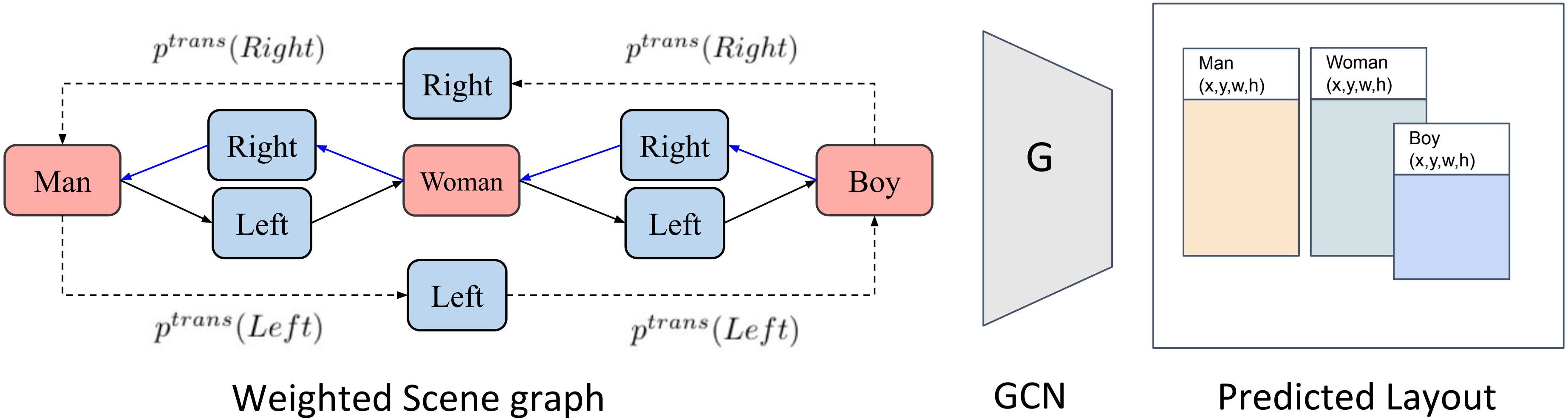


For every relation *r*, we learn the probability that it is transitive.

$$p^{trans}(r) = \sigma(\theta_r^{trans})$$

We complete transitive edges and assign them this probability

GCN for Weighted Scene Graph

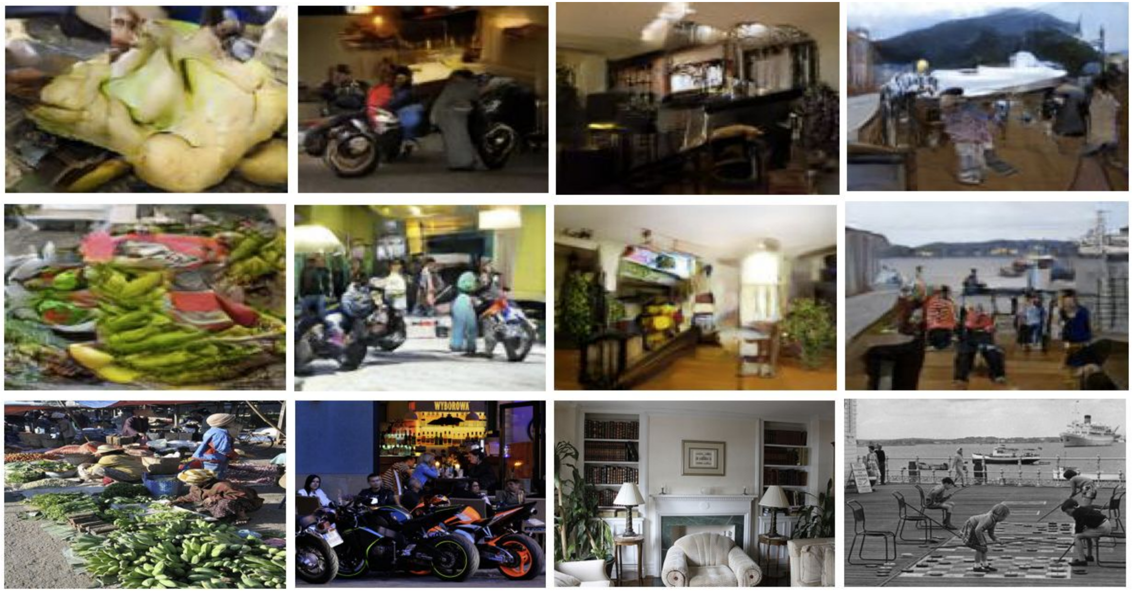


Generation Results

Sg2im*

Ours*

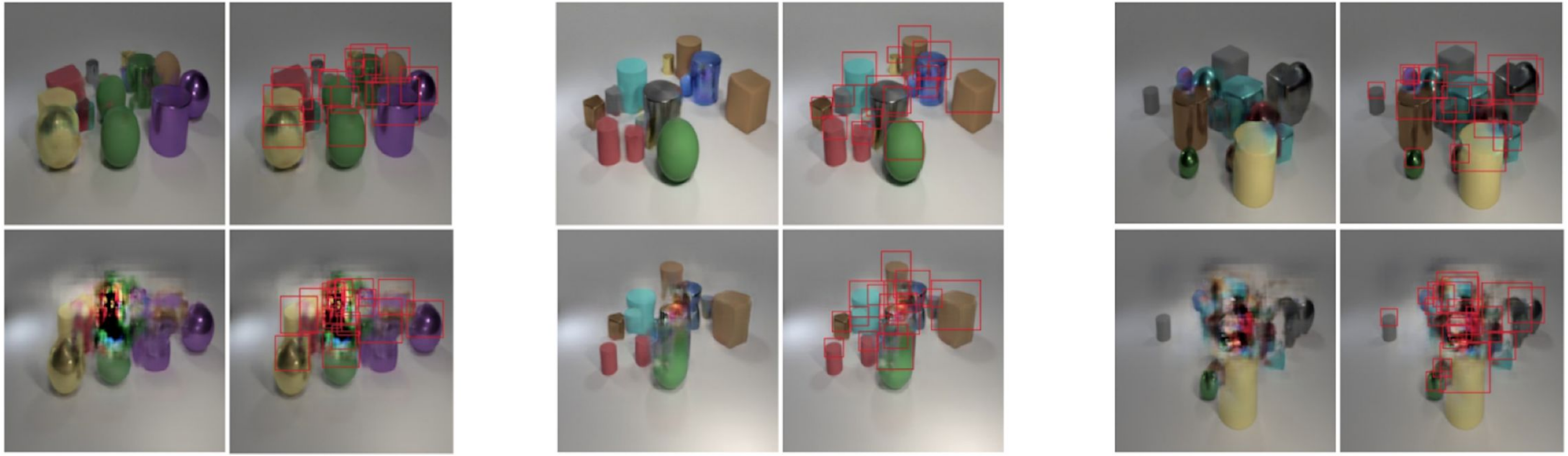
Ground Truth



* Only scene graph to layout component is compared. **Layout to image component is fixed** to LostGANs: Sun, W., & Wu, T. (2019). Image synthesis from reconfigurable layout and style. *ICCV 2019*

Large Graph Sizes

Improved performance over packed scenes



WSGC (ours)

Sg2Im (baseline)

Example #1 *Example #2* *Example #3*

Large Graph Sizes

Improved performance over packed scenes

Method	Standard						Packed					
	mIOU		R@0.3		R@0.5		mIOU		R@0.3		R@0.5	
	COCO	VG	COCO	VG	COCO	VG	COCO	VG	COCO	VG	COCO	VG
Sg2Im [17] 5 GCN^9	-	-	52.4	21.9	32.2	10.6	-	-	-	-	-	-
Sg2Im [17] 5 GCN^{10}	41.7	16.9	62.6	24.7	37.5	9.7	35.8	25.4	56.0	36.2	25.3	15.8
Sg2Im [17] 8 GCN^{10}	41.5	**18.3**	62.9	**26.2**	38.1	10.6	37.2	25.8	58.6	36.9	26.4	15.9
Sg2Im [17] 16 GCN^{10}	40.8	16.4	61.4	23.3	36.6	7.8	37.7	27.1	60.3	39.0	26.6	17.0
WSGC 5 GCN (ours)	**41.9**	18.0	**63.3**	25.9	**38.2**	10.6	**39.3**	**28.5**	**62.6**	**42.4**	**30.1**	**18.3**

Thank you!

Project page: https://roeiherz.github.io/CanonicalSg2Im/

Poster **#5328**