NetGAN
Generating Graphs via Random Walks

Aleksandar Bojchevski*, Oleksandr Shchur*, Daniel Zügner*, Stephan Günnemann
Technical University of Munich, Germany

ICML 2018

* Equal contribution

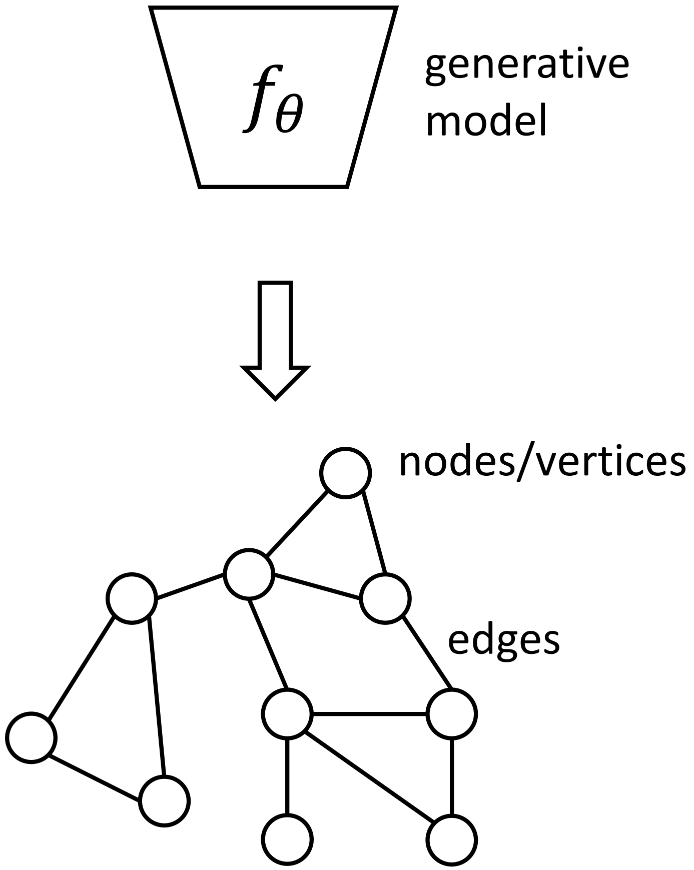
Generative models for graphs

Graphs are ubiquitous

- Social networks
- Protein-protein interaction networks
- Knowledge graphs

Generative models for graphs

- Imputation of missing values (link prediction, recommendation)
- Sampling / data simulation (planning, augmentation)
- Controllable generation
- Representation learning



Generative models for graphs

What makes a good generative model for graphs?

Should capture essential observed patterns of real-world graphs!

	Sparsity	Power law degree dist.	Community structure	Triangle counts	Homophily	Unknown patterns
Planted Part.	X		X		X	???
DC-SBM	X	X	X		X	???
ERGM	X	X		X	X	???
…						???
Super model	X	X	X	X	X	???

Generative models for graphs

What makes a good generative model for graphs?

Should capture essential observed patterns of real-world graphs!

	Sparsity	*Power law degree dist.	Community structure	Triangle counts	#Homophily	Unknown patterns
Planted Part.	X	???	X		???	???
DC-SBM	X	???	X		???	???
ERGM	X	???		X	???	???
…		???			???	???
Super model	X	???	X	X	???	???

*Broido & Clauset 2018, #Dong et al. 2017

Generative models for graphs

What makes a good generative model for graphs?

Sh

P

How do we define a model that captures all the essential (potentially still unknown) properties of real graphs?

Learn them from data ⇒ Implicit models

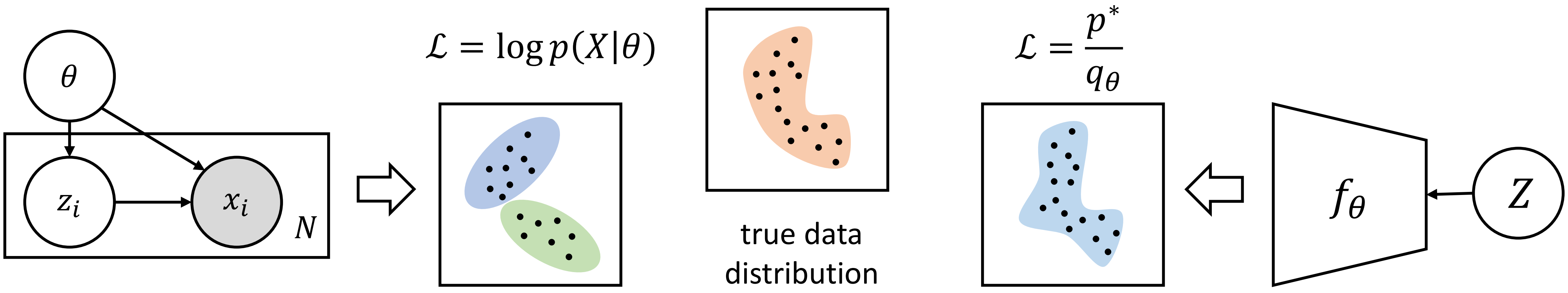
Super model	X	???	X	X	???	???

*Broido & Clauset 2018, #Dong et al. 2017

Explicit vs. implicit models

- **Explicit** models have a parametric specification of the data distribution
- Observe patterns and manually specify a model to capture them
- Learn via MLE, MAP, …

- **Implicit** models define a stochastic process that directly generates data
- Likelihood free: learn by comparison with the true data distribution (e.g. class probability estimation, GANs)



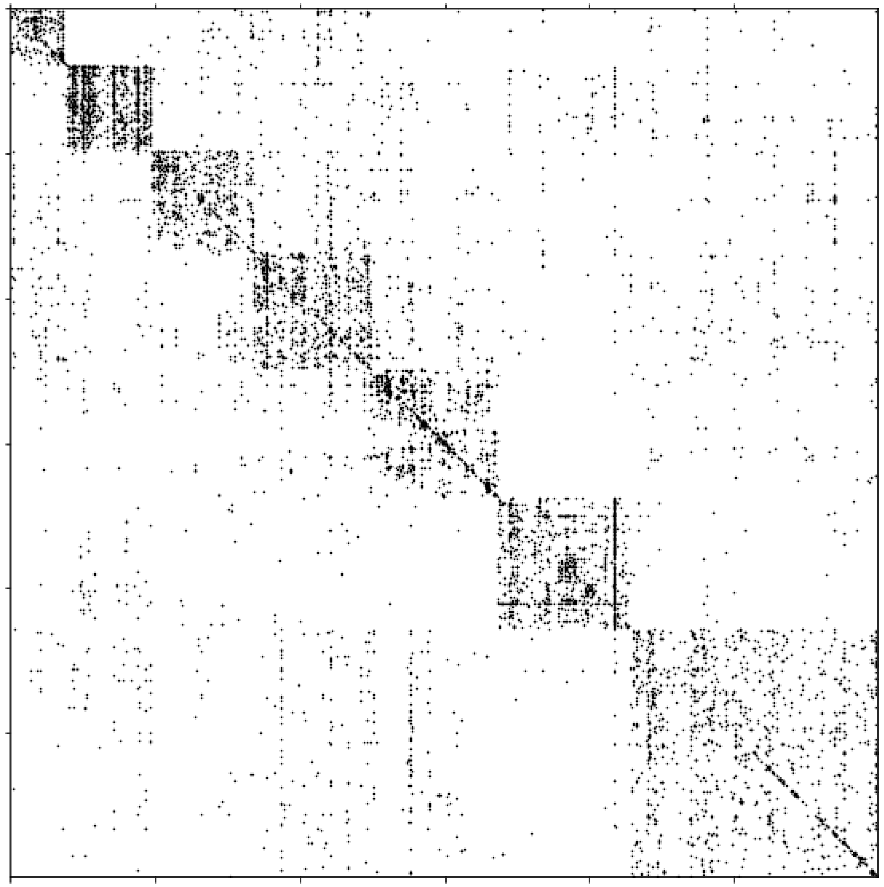
Setting and challenges

1. Single large graph as input
   - Compared to e.g. many images in computer vision

2. Quadratic scaling and sparsity
   - For N nodes there are N^2 possible edges
   - Real graphs have $|E| \ll N^2$ significantly fewer edges

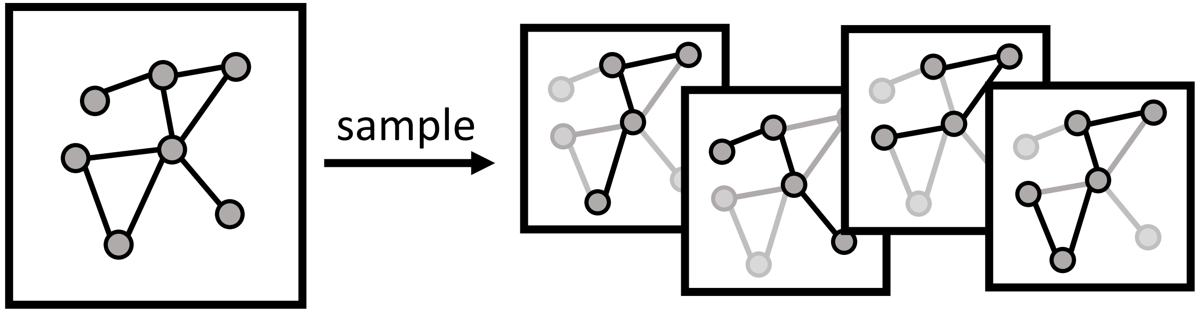
3. Discrete output samples
   - Can't easily backpropagate through sampling step

4. Permutation invariance

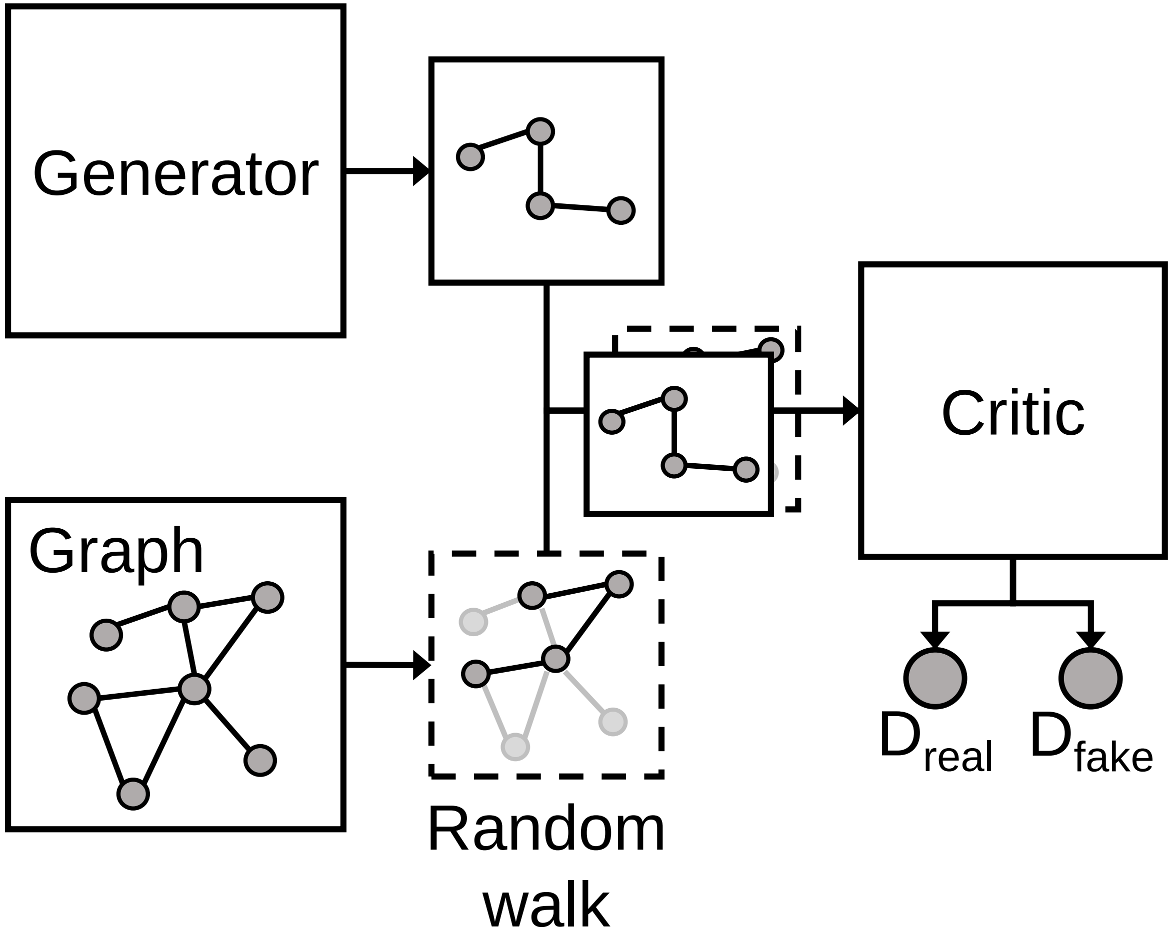


Tackling the challenges

Our solution: learn a distribution of random walks over the graph

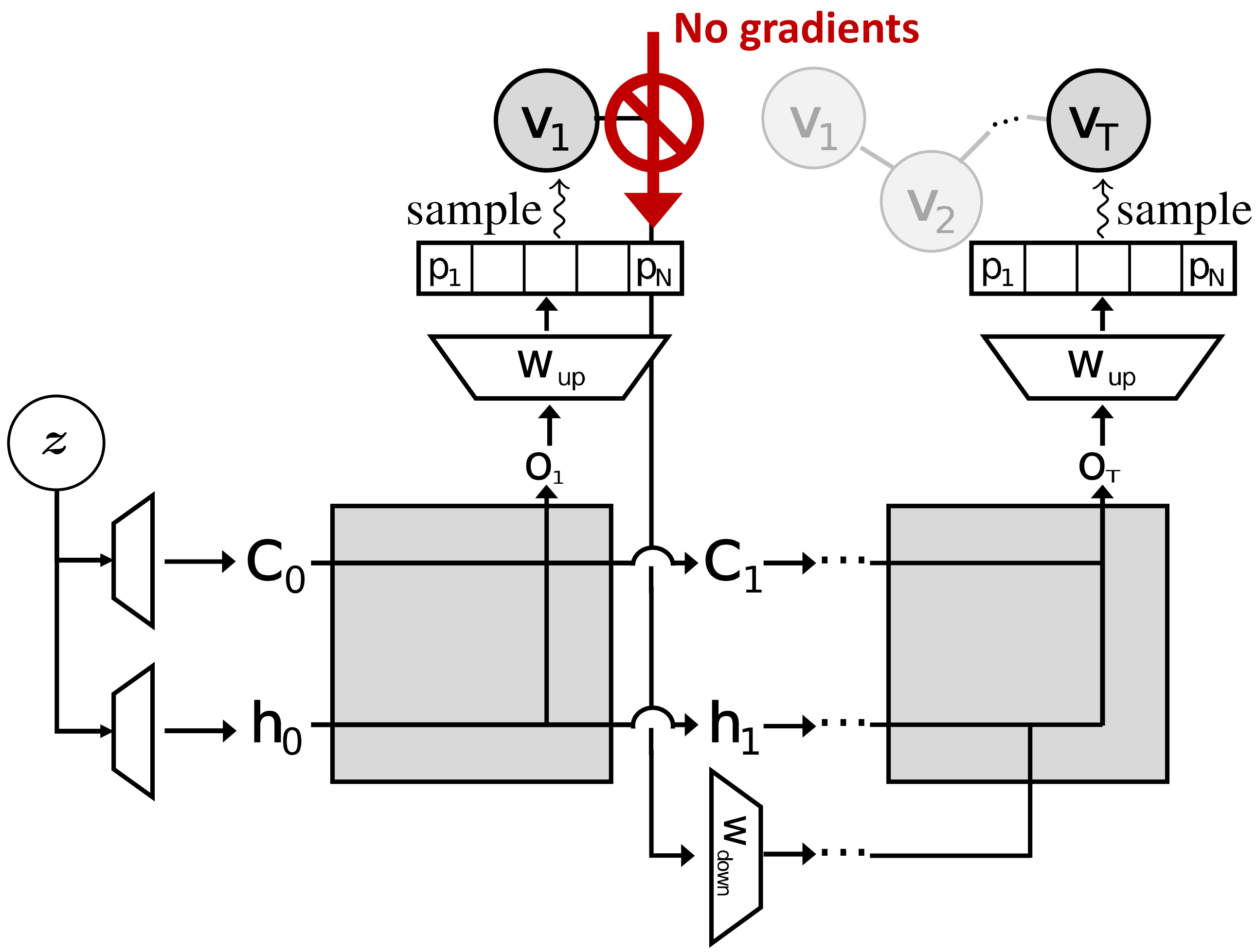


NetGAN architecture



Wasserstein GAN

NetGAN generator



$$\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$$

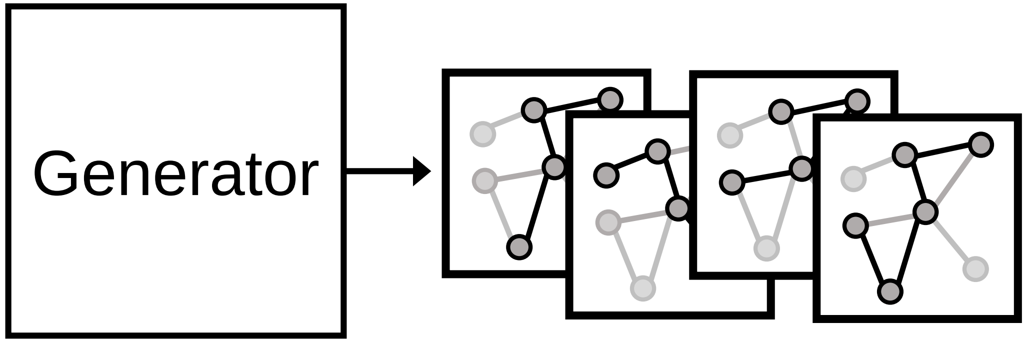
$$\boldsymbol{m}_0 = g_{\theta'}(\boldsymbol{z})$$

$$(\boldsymbol{p}_t, \boldsymbol{m}_t) = f_\theta(\boldsymbol{m}_{t-1}, \boldsymbol{v}_{t-1})$$

$$\boldsymbol{v}_t \sim \mathrm{Cat}(\sigma(\boldsymbol{p}_t))$$

Tackling discreteness:
Gumbel-Softmax straight-through estimator (Jang et al., 2017)

Graph assembly



sum →

0	0	24	0
0	0	11	3
29	7	0	18
0	0	14	0

sum of transition counts

sample →

0	0	1	0
0	0	1	0
1	1	0	1
0	0	1	0

adjacency matrix

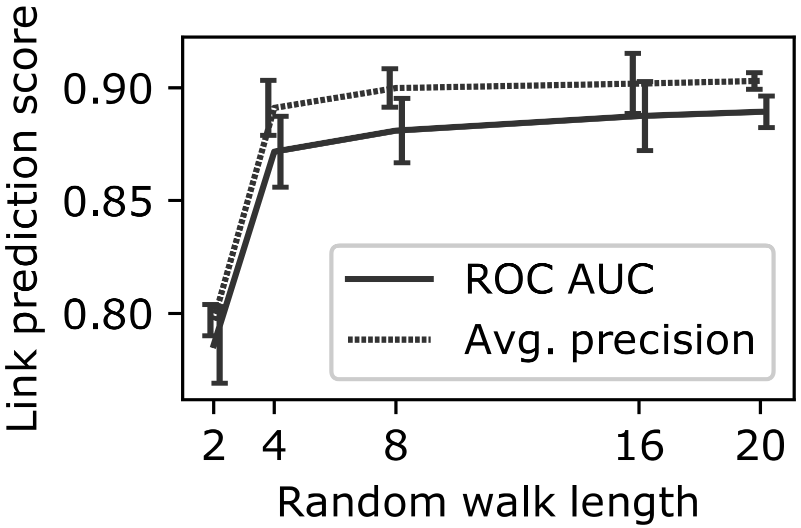
Graph assembly: sample edges with probability proportional to their transition counts

Random walk length

Random walks are a Markov process – they don't have memory

What's the benefit of having length greater than 2?

Helps with generalization (empirically)



Evaluation and goals

Unlike images we cannot visually inspect large graphs for quality

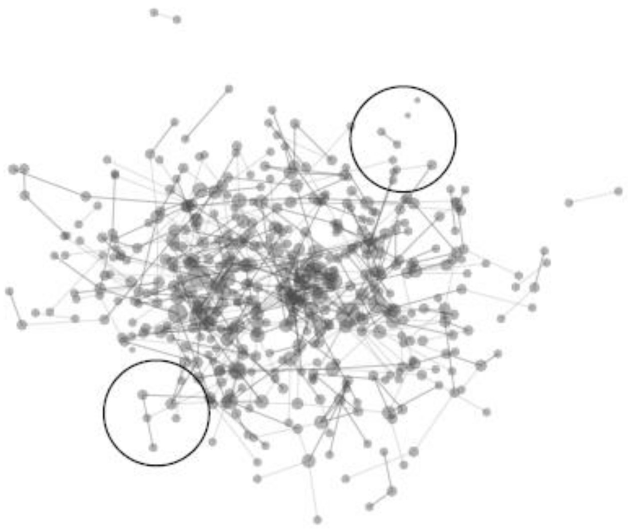
Goal 1) Fidelity - capture known graph patterns without manually specifying them

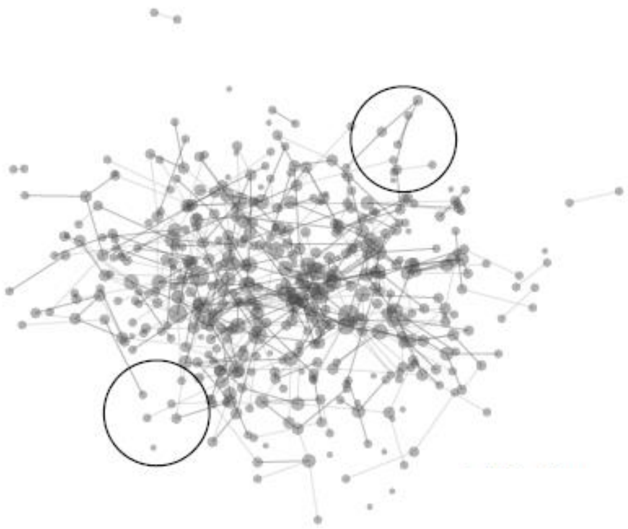
Goal 2) Generalization – capture the general topology
- Avoid trivial solution of memorizing training graph
- Impute missing data

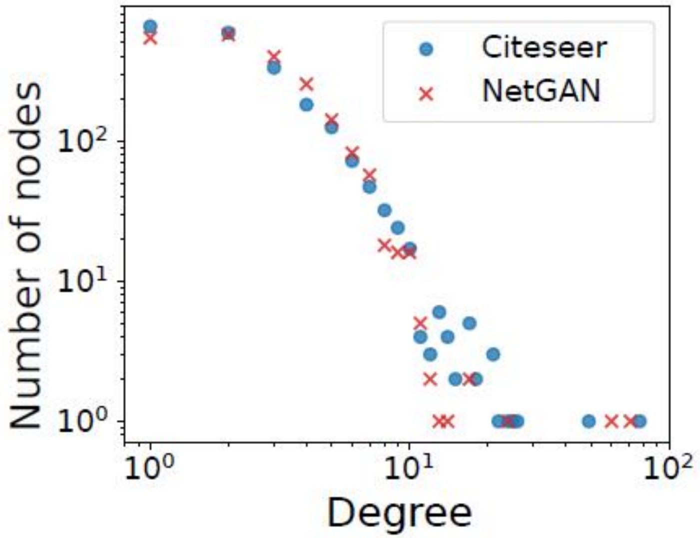
Goal 3) Latent space interpolation
- Yields graphs with smoothly changing properties

What can we do with NetGAN?

Generate graphs that have similar structure but are not replicas







Reproduce known patterns

… without manually specifying them

	Max degree	Power law exp.	Intra-Com. density	Inter-Com. density	Clustering coefficient	Character. path len.
Original graph	240	1.86	1.7e-3	4.3e-4	2.7e-3	5.61
Configuration model	240	1.86	2.8e-4	1.6e-3	3.0e-4	4.38
DC-SBM	165	1.81	1.2e-3	6.7e-4	3.3e-3	5.12
ERGM	243	1.79	1.2e-3	6.9e-4	2.2e-3	4.59
BTER	199	1.79	7.5e-4	1.0e-3	4.6e-3	4.59
VGAE	13	1.67	3.2e-4	1.4e-3	1.2e-3	5.28
NetGAN	233	1.79	1.4e-3	6.0e-4	2.4e-3	5.20

Generalize – impute missing data

Idea: assess link prediction performance on held out data via transition counts

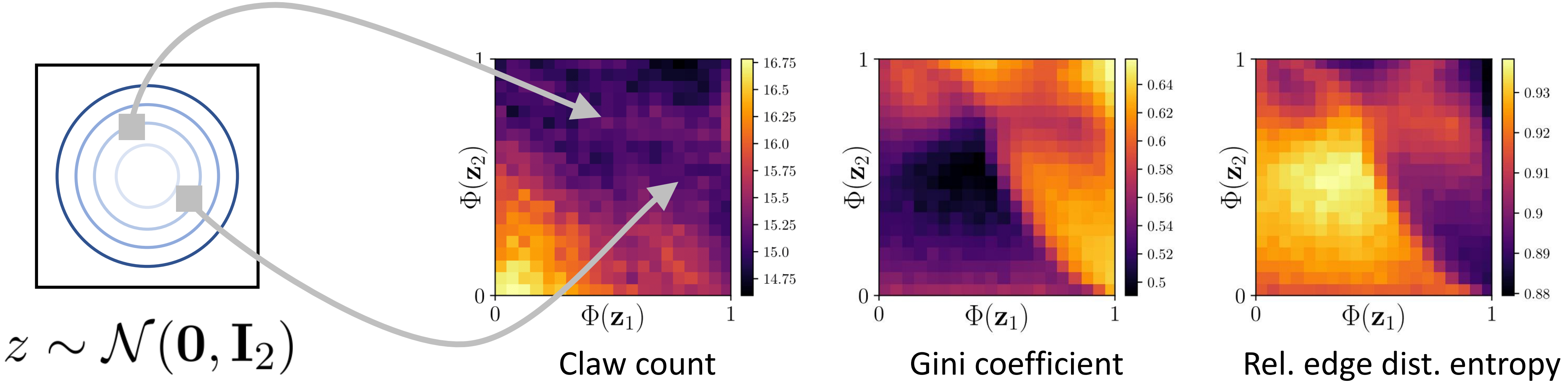
Higher transition count implies that the edge is more likely

	Cora-ML	Citeseer	Pubmed	PolBlogs	DBLP	Cora
number of nodes	$2.8\ K$	$2.1\ K$	$19.7\ K$	$1.8\ K$	$16.2\ K$	$18.8\ K$
number of edges	$7.9\ K$	$3.7\ K$	$44.3\ K$	$16.7\ K$	$51.9\ K$	$64.5\ K$
Adamic/Adar	92.16	88.69	84.98	85.43	91.13	93.00
DC-SBM	**96.03**	94.77	**96.76**	95.46	**97.05**	98.01
node2vec	92.19	95.29	96.49	85.10	96.41	**98.52**
VGAE	95.79	95.11	94.50	93.73	96.38	97.59
NetGAN (500K)	94.00	95.18	87.39	95.06	82.45	82.31
NetGAN (100M)	95.19	**96.30**	93.41	**95.51**	86.61	84.82

Interpolate between graphs

Instead of sampling from the entire 2D latent space, sample from small sub-regions

Each pixel in the plots corresponds to a graph (statistic) sampled from a sub-region



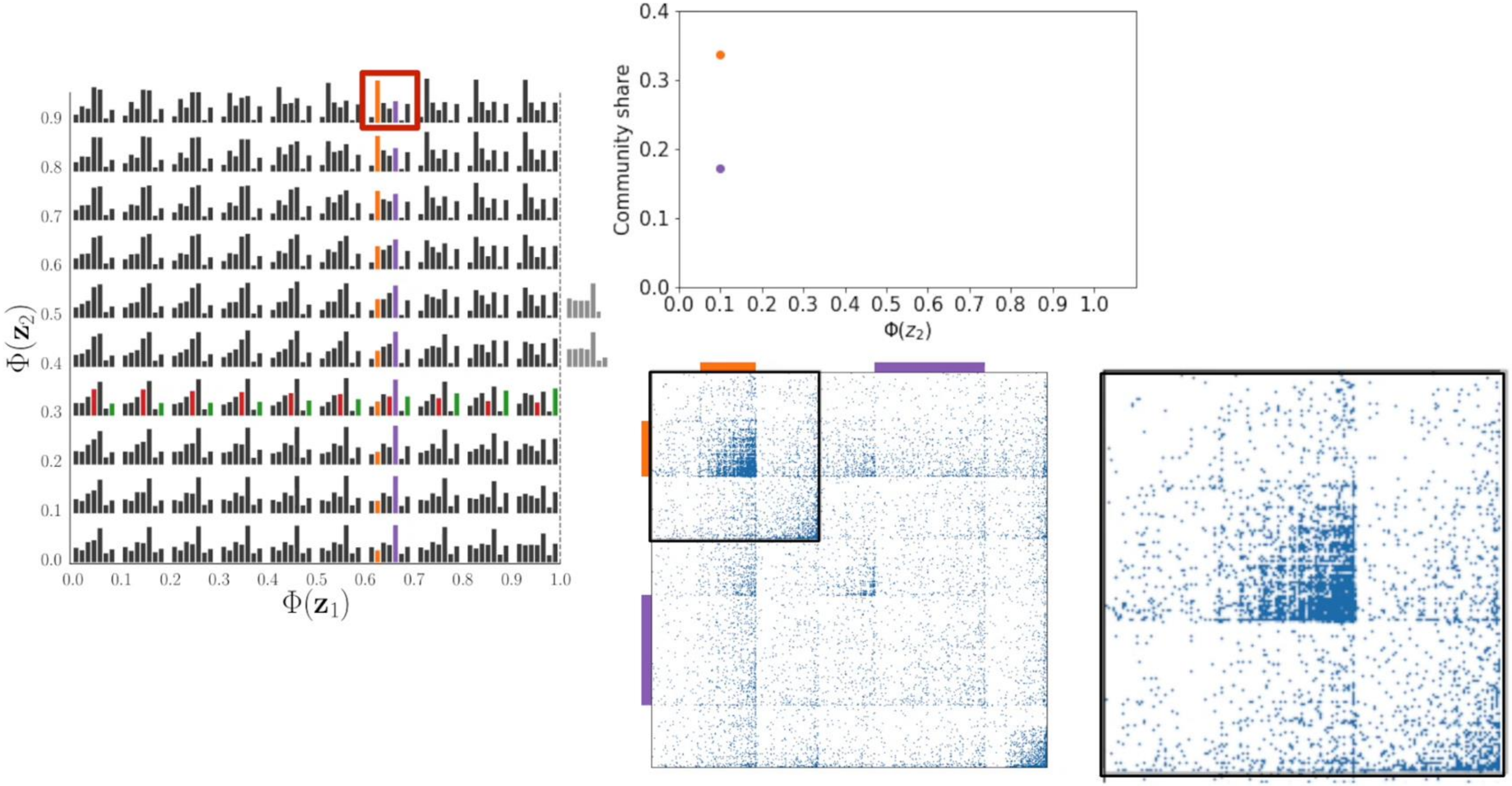
Claw count

Gini coefficient

Rel. edge dist. entropy

Interpolate between graphs

The density of communities also changes smoothly



Limitations and future work

Fixed graph size

- Can only generate graphs of same size as the input

Scalability

- Larger graphs need more random walks to get representative transition counts

Evaluation

- How to evaluate the quality beyond graph statistics

Other settings: heterogenous, attributed, multi-graph, ….

Summary

NetGAN – Deep implicit model generating graphs via random walks

Captures (un?)known graph patterns without manually specifying them

Generalizes as shown by its link prediction performance

Generates graphs with smoothly changing properties

Code: github.com/danielzuegner/netgan

Poster: #58, Hall B, Today