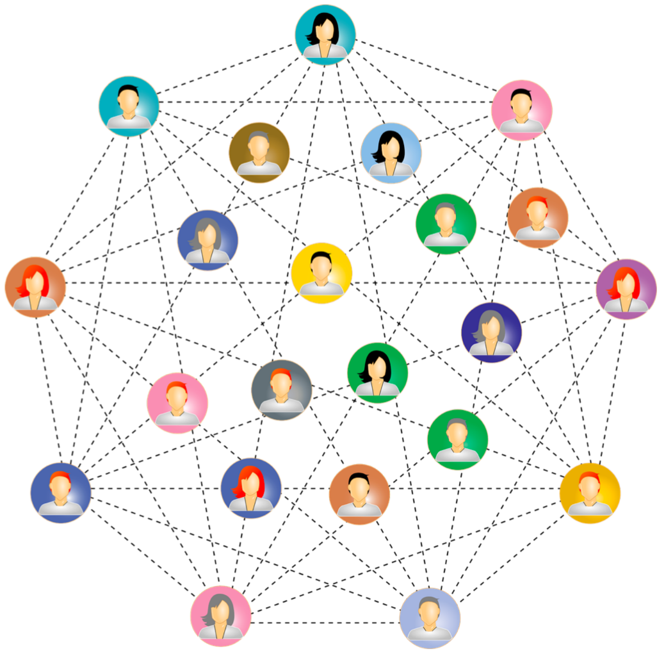
Graphite: Iterative Generative Modeling of Graphs

Aditya Grover, Aaron Zweig, Stefano Ermon
Computer Science Department
Stanford University

Graphs are ubiquitous



Social, biological, information networks etc.

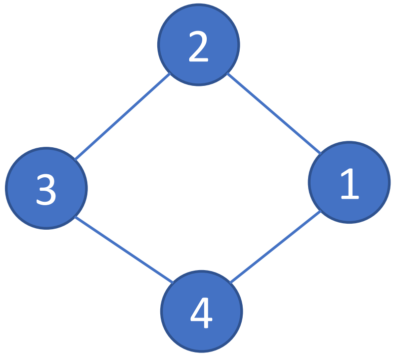
How do we **learn representations** of nodes in a graph?

Useful for several prediction tasks. *E.g.*, friendship links on social networks (**link prediction**), living status of organisms in ecological networks (**node classification**)

Latent Variable Model of a Graph

- Graphs are represented as adjacency matrices $\mathrm{A} \in \{0,1\}^{n \times n}$
- For every node i, we associate a latent vector representation $\mathbf{z}_i \in \mathbb{R}^k$

Example graph



Adjacency matrix

$$\mathrm{A} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Latent feature matrix

$$\mathrm{Z} = \begin{pmatrix} \mathbf{z}_1^T \\ \mathbf{z}_2^T \\ \mathbf{z}_3^T \\ \mathbf{z}_4^T \end{pmatrix}$$

Graphite: A VAE for Graphs

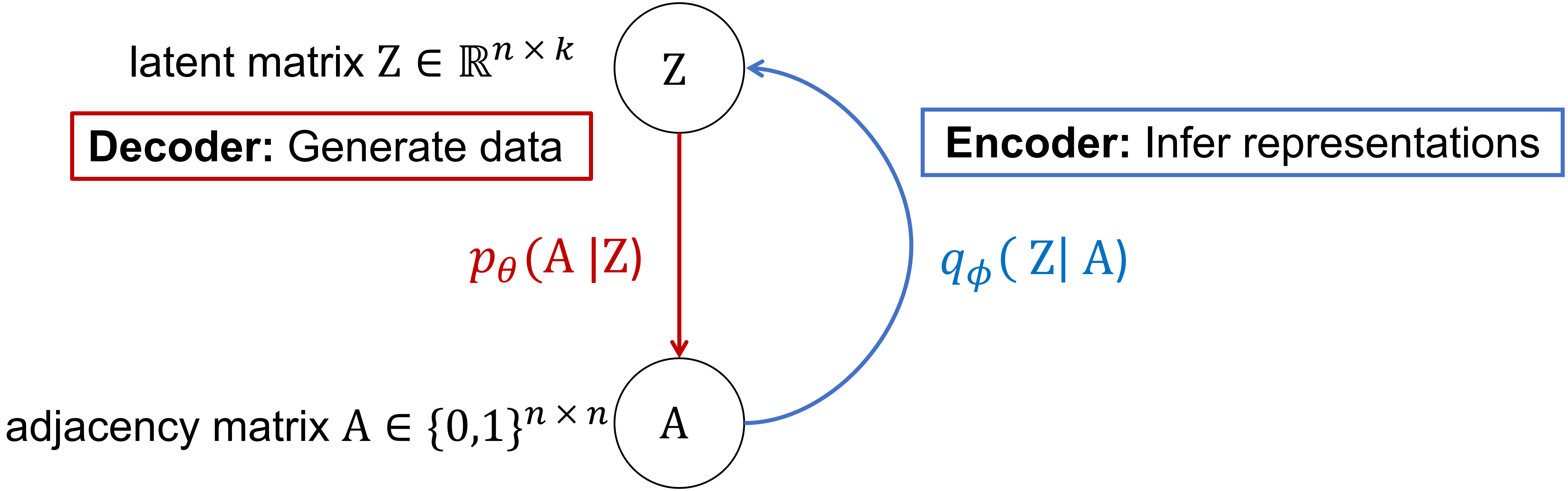
latent matrix $\mathrm{Z} \in \mathbb{R}^{n \times k}$

Decoder: Generate data

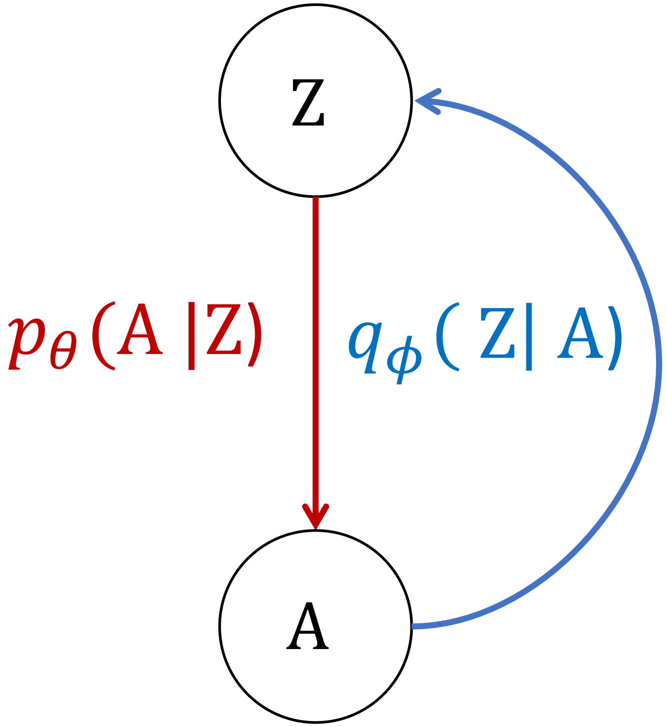
$p_\theta(\mathrm{A} \mid \mathrm{Z})$

adjacency matrix $\mathrm{A} \in \{0,1\}^{n \times n}$

Graphite: A VAE for Graphs

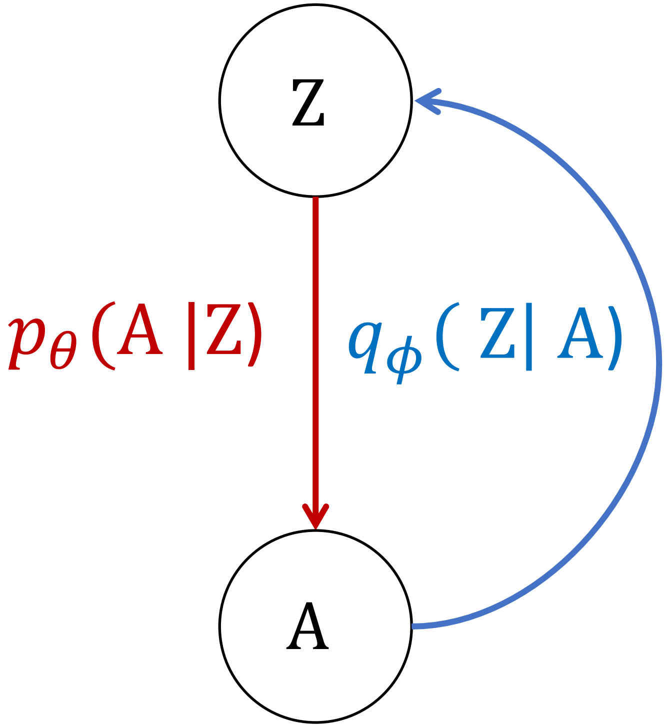


Graphite: Learning & Inference



Given: Dataset of adjacency matrices, D_A

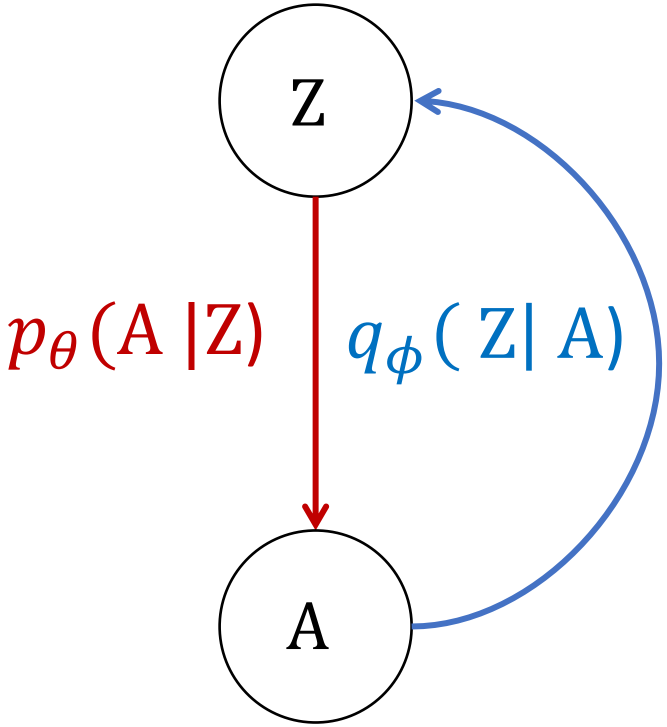
Graphite: Learning & Inference



Given: Dataset of adjacency matrices, D_A

Learning objective: $\max_{\theta,\phi} \mathrm{ELBO}(\theta, \phi; \mathrm{D}_A)$

Graphite: Learning & Inference



Given: Dataset of adjacency matrices, D_A

Learning objective: $\max_{\theta,\phi} \mathrm{ELBO}(\theta, \phi; \mathrm{D}_A)$

Test time use cases

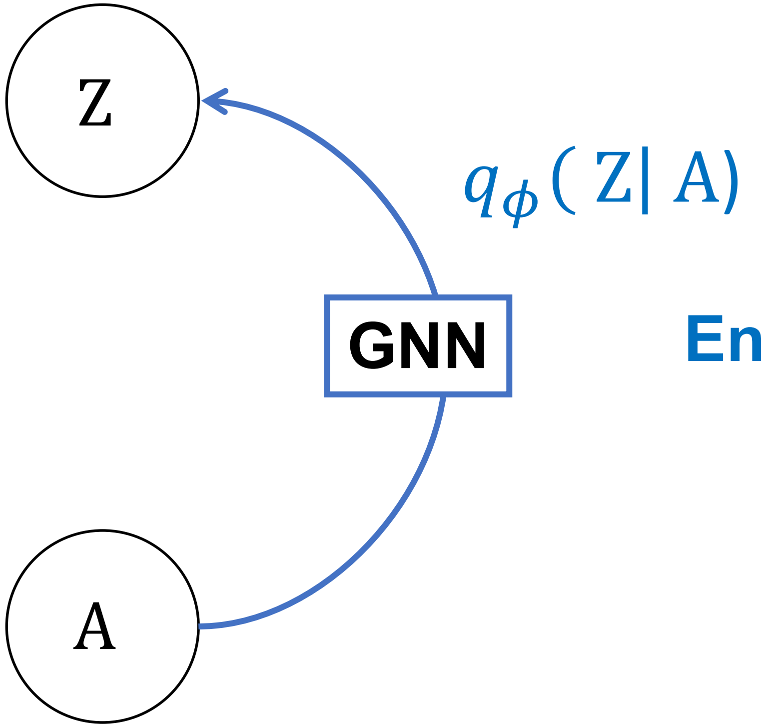
Generative modeling tasks

- Density estimation, clustering nodes, compressing graphs etc.

Graph tasks

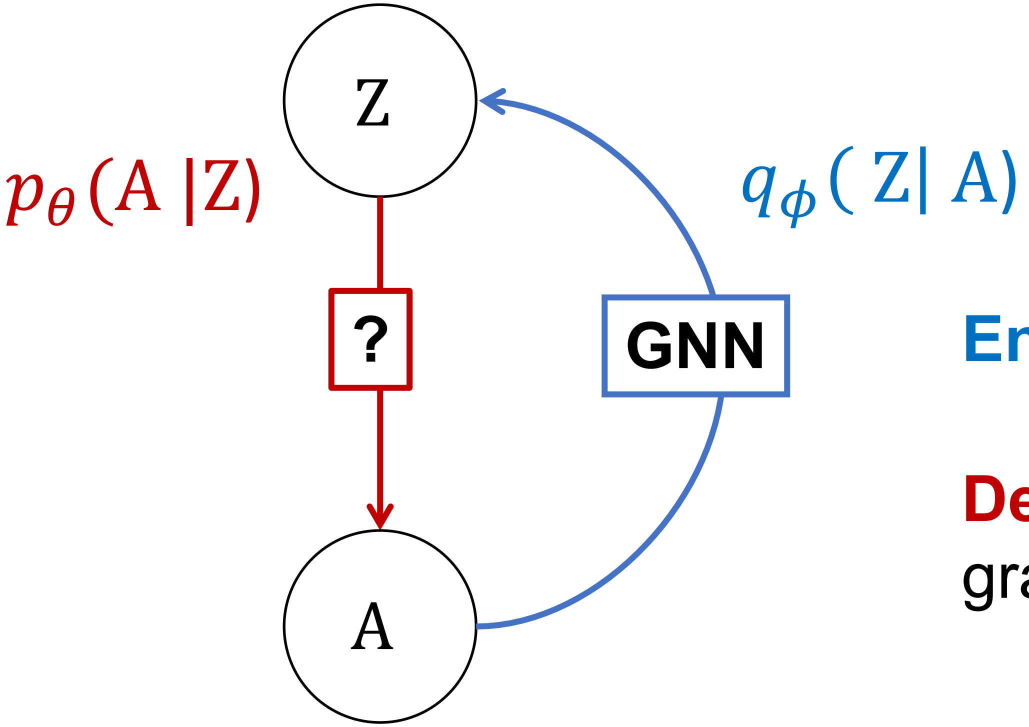
- *Link Prediction:* Denoise graph
- *Semi-supervised node classification:* Feed $\mathbf{z}_i$ for labelled nodes to a classifier

Parameterizing Graph Autoencoders



Encoding $q_\phi(Z|A)$: Graph Neural Network (GNN)

Parameterizing Graph Autoencoders



Encoding $q_\phi(\mathrm{Z}|\,\mathrm{A})$: Graph Neural Network (GNN)

Decoding $p_\theta(\mathrm{A}\,|\mathrm{Z})$: Challenging to “upsample” graphs given latent representations

Decoding Graphs - MLP

$Z \in \mathbb{R}^{n \times k}$

$p_\theta(A \mid Z)$

$A \in \{0,1\}^{n \times n}$

Option 1: Multi-layer Perceptrons (MLP)

Simonovsky et al., 2018

$O(n^2 d + dk)$ total parameters for single hidden layer of width d

Decoding Graphs - RNN

$Z \in \mathbb{R}^{n \times k}$

$p_\theta(A\,|Z)$

$A \in \{0,1\}^{n \times n}$

Option 2: Recurrent Neural Network (RNN)

You et al., 2018

Arbitrary ordering of nodes
required for training
e.g., BFS, DFS

Graphite – Decoding Graphs using GNN

$Z \in \mathbb{R}^{n \times k}$

$p_\theta(A\,|Z)$

$A \in \{0,1\}^{n \times n}$

Key idea
Learn the low-rank structure of adjacency matrix A in the latent space Z

Graphite – Decoding Graphs using GNN

$Z \in \mathbb{R}^{n \times k}$

$p_\theta(A \mid Z)$

$A \in \{0,1\}^{n \times n}$

- For fixed number of iterations:
  Step 1 (Low rank matrix reconstruction)
  Map Z to an intermediate graph $\widehat{A}$ via an inner product
  $\widehat{A} \approx ZZ^T$

Graphite – Decoding Graphs using GNN

$Z \in \mathbb{R}^{n \times k}$

$p_\theta(A \mid Z)$

$A \in \{0,1\}^{n \times n}$

- For fixed number of iterations:

  Step 1 (Low rank matrix reconstruction)

  Map Z to an intermediate graph $\hat{A}$ via an inner product

  $$\hat{A} \approx ZZ^T$$

  Step 2 (Progressive refinement)

  Refine Z by message passing over $\hat{A}$ using a GNN

  $$Z = \mathrm{GNN}_\theta(\hat{A})$$

Graphite – Decoding Graphs using GNN

$\mathrm{Z} \in \mathbb{R}^{n \times k}$

$p_\theta(\mathrm{A} \mid \mathrm{Z})$

$\mathrm{A} \in \{0,1\}^{n \times n}$

- For fixed number of iterations:

  Step 1 (Low rank matrix reconstruction)

  Map Z to an intermediate graph $\hat{\mathrm{A}}$ via an inner product

  $$\hat{\mathrm{A}} \approx \mathrm{ZZ}^{\mathrm{T}}$$

  Step 2 (Progressive refinement)

  Refine Z by message passing over $\hat{\mathrm{A}}$ using a GNN

  $$\mathrm{Z} = \mathrm{GNN}_\theta(\hat{\mathrm{A}})$$

- **Output step:** Set $p_\theta(\mathrm{A} \mid \mathrm{Z})$ = Bernoulli(sigmoid(ZZ^{T}))

Graphite – Decoding Graphs using GNN

$Z \in \mathbb{R}^{n \times k}$

$p_\theta(A \mid Z)$

$A \in \{0,1\}^{n \times n}$

- **Unlike MLP**, GNN decoder with single hidden layer of length *d* has $O(\mathrm{dk})$ parameters
- **Unlike RNN**, no arbitrary ordering of input nodes is required

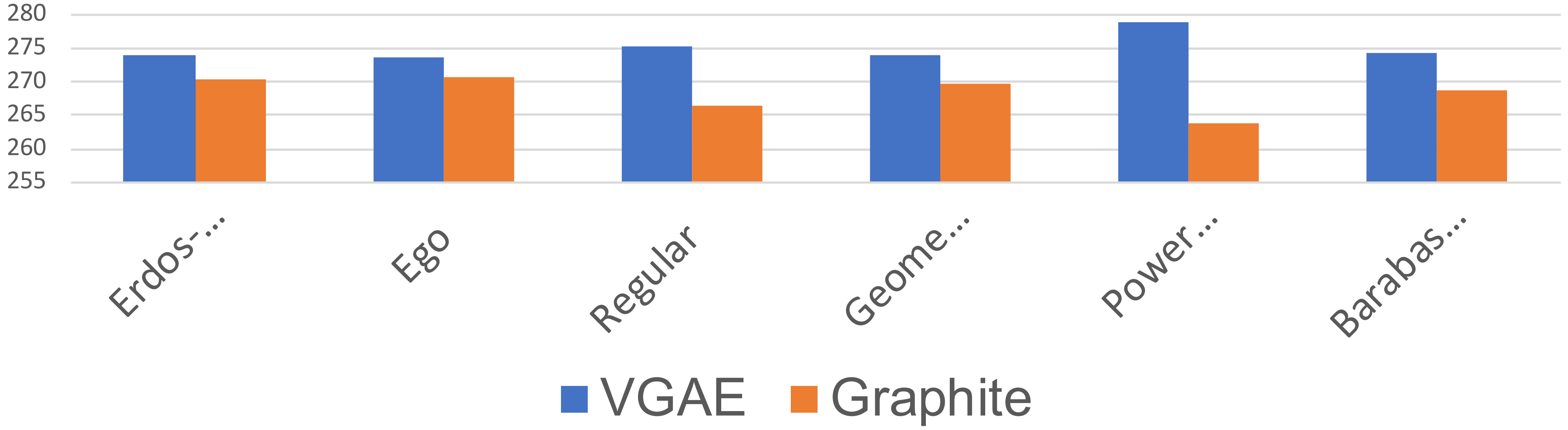
Decoding is also **computationally** efficient. See paper for details.

Empirical Results – Density Estimation

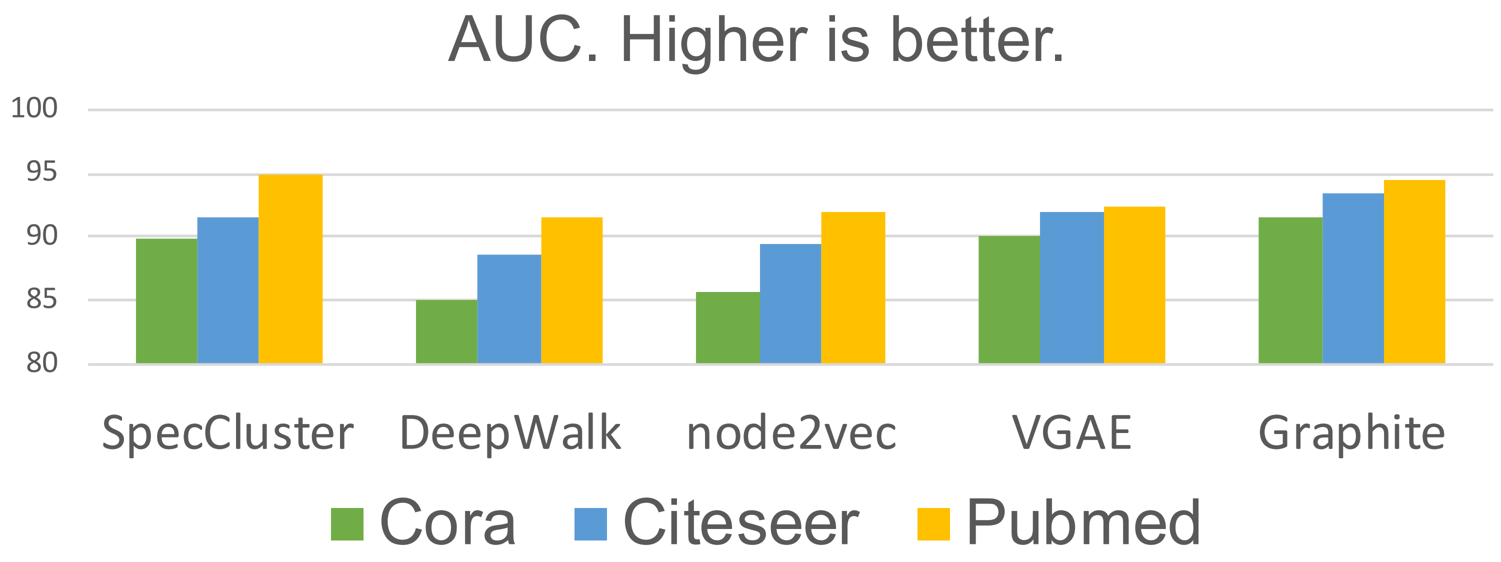
Baseline VGAE [Kipf et al., 2016]

GNN Encoder + Non-learned Inner Product Decoder. No iterative refinement.

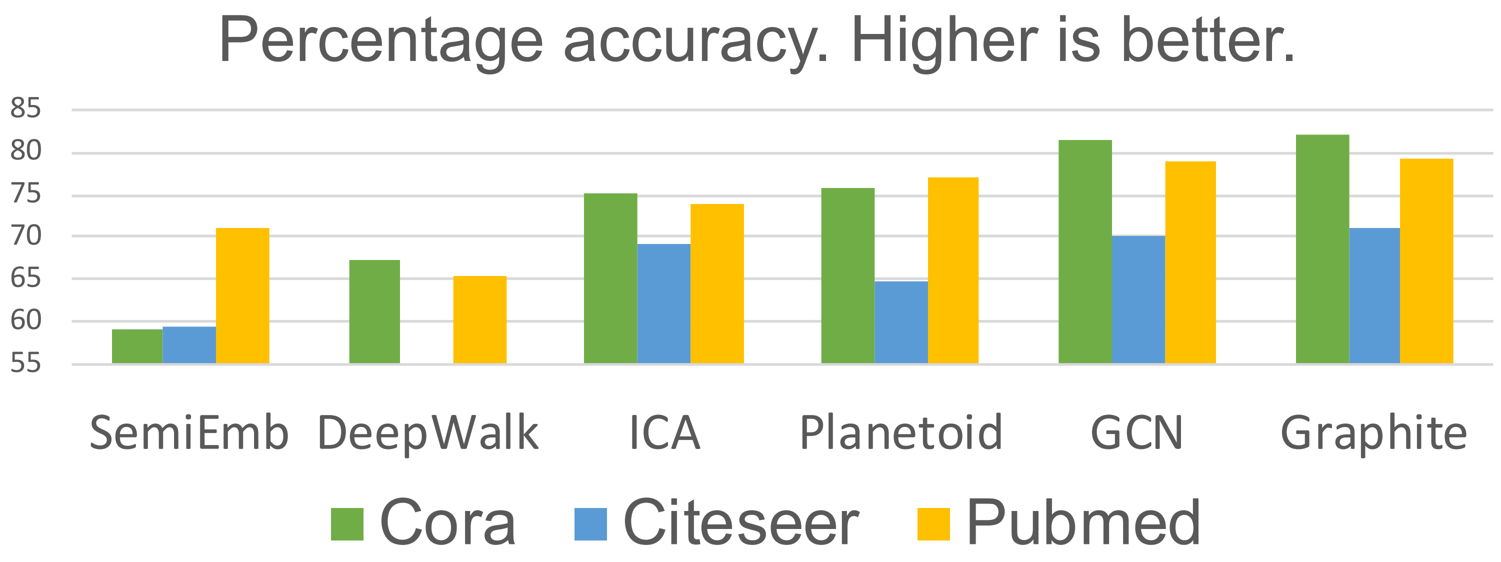
Negative log-likelihoods. Lower is better.



Empirical Results – Link Prediction



Empirical Results – Semi-supervised Node Classification



Summary

Graphite: A latent variable generative model for graphs where both encoder and decoder are parameterized by graph neural networks.

- **Encoder** performs message passing on input graph
- **Decoder** iteratively refines inner product graphs

For more details, please visit us at Poster #7.
Code: https://github.com/ermongroup/graphite