The Power of Summarization in Network Representation Learning

Danai Koutra
Assistant Professor, CSE
Computational Medicine and Bioinformatics (courtesy)


Great Lakes Workshop on Data Science – September 20-22, 2019

Slides at: https://bit.ly/2m2tlJo
Welcome!

We are the Graph Exploration and Mining at Scale (GEMS) lab at the University of Michigan, founded and led by Danai Koutra. Our team researches important data mining and machine learning problems involving interconnected data, in other words, graphs or networks.

From airline flights to traffic routing to neuronal interactions in the brain, graphs are ubiquitous in the real world. Their properties and complexities have long been studied in fields ranging from mathematics to the social sciences. However, many pressing problems involving graph data are still open. One well-known problem is scalability. With continual advances in data generation and storage capabilities, the size of graph datasets has dramatically increased, making scalable graph methods indispensable. Another is the changing nature of data. Real graphs are almost always dynamic, evolving over time. Finally, many important problems in the social and biological sciences involve analyzing not one but multiple networks.

So, what do we do?

The problems described above call for principled, practical, and highly scalable graph mining methods, both theoretical and application-oriented. As such, our work covers fields like linear algebra, distributed systems, deep learning, and even neuroscience. Our ongoing projects include:

- Algorithms for multi-network tasks, like matching nodes across networks
- Learning low-dimensional representations of networks in metric spaces
- Abstracting or “summarizing” a graph with a smaller network
- Analyzing network models of the brain derived from fMRI scans
- Distributed graph methods for iteratively solving linear systems
- Network-theoretical user modeling for various data science applications

We’re grateful for funding from Adobe, Amazon, the Army Research Lab, the MIDAS program for Data Science (MIDAS), Microsoft Azure, the National Science Foundation (NSF), and the Army Research Lab.

Interested?

If you’re interested in joining our group, send an email with your interests and CV to opportunities@umich.edu.
A lot of work on network representation learning!

Must-read papers on NRL/NE.

NRL: network representation learning, NE: network embedding.

Contributed by Cunchao Tu, Yuan Yao and Zhengyan Zhang.

We release OpenNE, an open source toolkit for NE/NRL. This repository’s Representation Learning (training and testing framework. Currently, the DeepWalk, LINE, node2vec, GraRep, TADW and GCN.

Survey papers:
1. Representation Learning on Graphs: Methods and Applications. Yi 2017. paper
3. A Comprehensive Survey of Graph Embedding: Problems, Techni

55. Generative Adversarial Network based Heterogeneous Bibliographic Network Representation for Personalized Citation Recommendation. J. Han, Xiaoyan Cai, Libin Yang. AAAI 2018.
57. Structural Deep Embedding for Hyper-Networks. Kl paper
58. TIMERS: Error-Bounded SVD Restart on Dynamic Ni Zhu. AAAI 2018. paper
60. Bernoulli Embeddings for Graphs. Vinith Misra, Sum paper
61. Distance-aware DAG Embedding for Proximity Sean Zhuo, Fanwei Zhu, Kevin Chen-Chuan Chang, Zhao Zhou. AAAI 2018. paper code
62. GraphGAN: Graph Representation Learning with Ge, Miao ZHAO, Weinan Zhang, Fu Zheng, Zhang, Zhi Wang, MIAO ZHAO, Weinan Zhang, Fu Zheng, Zhang. AAAI 2018. paper
63. HARP: Hierarchical Representation Learning for Net AAAI 2018. paper code
64. Representation Learning for Scale-free Networks. R paper
65. Social Rank Regulated Large-scale Network Embed 2018. paper
66. Integrative Network Embedding via Deep Joint Reconstruction. Di Jin, Meng Ge, Liang Yang, Dongxiao He, paper
67. Scalable Multiplex Network Embedding. Hongming Zhang, Liwei Qiu, Lingling Yi, Yangqiu Song. UCI 2018. paper
68. Adversarially Regularized Graph Autoencoder for Graph Embedding. Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, Chengqi Zhang. UCI 2018.
72. Active Discriminative Network Representation Learning. Li Gao, Hong Yang, Chuan Zhou, Jia Wu, Shirui Pan, Yue Hu. UCI 2018.
75. Constructing Narrative Event Evolutionary Graph for Script Event Prediction. Zhongyang Li, Xiao Ding, Ting Liu. UCI 2018. paper code

https://github.com/thunlp/NRLPapers
A lot of work on network representation learning!
A lot of work on network representation learning!

Most work preserves proximity between nodes.
Proximity vs. Structural Similarity

Find similar nodes in the **same part** of the network

Useful for link prediction, clustering, classification assuming **homophily**

[Perozzi+ ’14; Grover+ ’16; Tang+ ’15; …]

Find nodes with similar roles **all over** the network

Useful for **role-based** classification, transfer learning, …

[Henderson+. KDD ‘12]

[Ribeiro+ ‘17; Donnat+ ’18, ..]
What are roles?

- The ways in which nodes / entities / actors relate to each other
- “The behavior expected of a node occupying a specific position” [Homans ‘67]
  - e.g., centers of stars
  - members of cliques
  - peripheral nodes
- Position or equivalence class:
  - collection of nodes with the same role

Network Science Co-authorship Graph [Newman 2006]
Relevant Sociology Literature

- D.R. White, K. Reitz: Graph and Semi-Group Homomorphism on Networks and Relations. Social Networks, 5:143-234, 1983.
Sometimes structural similarity is more appropriate than proximity.

Alignment or matching [CIKM’18]

Node classification [KDD’19c]

Anomaly detection [KDD’19b]

Graph comparison / classification [KDD’19a; ICDM’19a]

Role query

Identity resolution [PKDD’19]

Transfer learning
Sometimes structural similarity is more appropriate than proximity

From Community to Role-based Graph Embeddings
Ryan A. Rossi, Di Jin, Sungchul Kim, Nesreen K. Ahmed, Danai Koutra, John Boaz Lee

arXiv
https://arxiv.org/abs/1908.08572
This talk: Summarization in Network Representation Learning

• Summarization within a GCN for faster training, data denoising and interpretability [ACM KDD’19a]

• Embedding summarization for compression and on-the-fly computation [ACM KDD’19b; PKDD’19]
This talk: Summarization in Network Representation Learning

- **Summarization within a GCN** for faster training, data denoising and interpretability [ACM KDD’19a]

- **Embedding summarization** for compression and on-the-fly computation [ACM KDD’19b; PKDD’19]
Interpretable NN-based Classification

• **Given** a set of **networks**
  ✦ each associated with a **label**

• **Devise** an **efficient, interpretable, and parsimonious** model
  ✦ that can accurately predict and
  ✦ explain each label

[Yujun Yan, Jiong Zhu, et al. ACM KDD ’19]
Interpretable NN-based Classification

- Given a set of networks \(\rightarrow\) fMRI brain graphs
  - each associated with a label \(\rightarrow\) phenotype

- Devise an efficient, interpretable, and parsimonious model
  - that can accurately predict and explain each label \(\rightarrow\) phenotype

[Yujun Yan, Jiong Zhu, et al. ACM KDD ’19] [https://github.com/GemsLab/GroupINN]
Related Work

• **Linear models** (PCA, ICA, matrix factorization)
  + Denoising
  - Fail to capture non-linear interactions

• **Neural-network models** (different variants of GCN)
  + Able to model non-linear interactions
  - Need many training samples
  - Need many parameters
  - Long time for training
  - “Black” box

<table>
<thead>
<tr>
<th>Model (Year)</th>
<th>Fast</th>
<th>Parsimonious</th>
<th>Interpretable</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN (KDD’17), GraphCNN (NIPS’16)</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>GCN (ICLR’17), DGCNN (AAAI’18)</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Diffpool (NIPS’18)</td>
<td>✓</td>
<td>✗</td>
<td>inadequate</td>
</tr>
<tr>
<td>GroupINN (proposed)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Related Work

• **Linear models** (PCA, ICA, matrix factorization)
  + Denoising
  −

• **Neural-network models** (different variants of GCN)
  + Able to model non-linear interactions between ROIs
  − Need many training samples
  − Need many parameters
  − Long time for training
  − “Black” box

Can we build an **interpretable NN-based model** that is **insensitive to noise**, **parsimonious** and able to capture **nonlinearities** in the prediction task?

<table>
<thead>
<tr>
<th>Model</th>
<th>Fast</th>
<th>Parsimonious</th>
<th>Interpretable</th>
<th>Adequate</th>
</tr>
</thead>
<tbody>
<tr>
<td>GraphCNN (NIPS’16)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>×</td>
</tr>
<tr>
<td>GCN (ICLR’17), DGCNN (AAAI’18)</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>×</td>
</tr>
<tr>
<td>Diffpool (NIPS’18)</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
<td>inadequately</td>
</tr>
<tr>
<td>GroupINN (proposed)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
GroupINN Architecture

[Correlation-based Network (with + & - weights)]
GroupINN Architecture

Graph Summarization to
• handle noisy data
• train from small samples of high-dim data
• support interpretability

GroupINN Architecture

GroupINN Architecture

1. Node Grouping / *Summarization* Layer

- Recent findings have shown that some nodes (ROIs) are most related to the phenotype of interest → some edges are expected to be more indicative
  
  [Cohen+J Neurosci ’16] [Cole+ NeuroImage ’07]

- Node grouping layer:
  - “hides” the non-indicative edges into a *supernode* and
  - highlights the indicative edges

---

1. Node Grouping / Summarization Layer

- **F**: learnable common membership matrix

Real valued **importance score** of node \( i \) in the prediction task

**Interpretability**
- Nonnegative
- Orthogonal (ideally)
- Nodes in supernode **not** required to be similar / well-connected

\[
\text{Adj. of supergraph } \mathbf{W}^s = \mathbf{F}^T \mathbf{W} \mathbf{F}
\]

Details
2. RWR-based Graph Conv Layer: Intuition

- **Random walks:**
  - useful tool to **sample graph structure**
  - the RWR scores quantify the **similarities** of other nodes to the seed nodes

- **Design:** The output $Y_i$ of layer $i$ is:
  $$Y_i = \sigma(cW^sY_{i-1}Q_i + I)$$

For more structure, multiple $\tilde{Q}$ at diff. distances $q_1 q_2 q_3$

[Yujun Yan, Jiong Zhu, et al. ACM KDD ’19] [https://github.com/GemsLab/GroupINN]
Q1. Comparison with NN-based methods

GroupINN models are up to 69× faster at training than all the baseline methods, while achieving same or higher accuracy in a variety of prediction tasks.

---

[Yujun Yan, Jiong Zhu, et al. ACM KDD ’19] [https://github.com/GemsLab/GroupINN]
Q2. Parsimony of GroupINN

Less is better!

<table>
<thead>
<tr>
<th>Methods</th>
<th># parameters</th>
<th>Normalized wrt GroupINN</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNN-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GCN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diffpool</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>GroupINN</strong></td>
<td>2,892</td>
<td><strong>1×</strong></td>
</tr>
</tbody>
</table>

thanks to the summarization layer

GroupINN can use 15% or much fewer model parameters to achieve comparable or better performance of the baseline methods.

Q3. Interpretability

GroupINN finds the most task-positive sub-networks.

PCA and Diffpool are misled by strong noisy signals from mouth and hand motion.

thanks to the summarization layer

[Acronyms of brain subnetworks. AN: auditory; CBLN: cerebellar; CON: cingulo-opercular; DAN: dorsal attention; FPN: frontoparietal; MRN: memory retrieval; SN: salience; VAN: ventral attention; VN: vision; SM.M: sensory/somatomotor mouth; SM.H: sensory/somatomotor hand]

**Tasks**

<table>
<thead>
<tr>
<th>Working Memory</th>
<th>Gambling</th>
<th>Emotion</th>
<th>Social</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRN</td>
<td>VAN</td>
<td>SN</td>
<td>FPN</td>
</tr>
<tr>
<td>FPN</td>
<td>VN</td>
<td>CON</td>
<td>SN</td>
</tr>
<tr>
<td>SN</td>
<td>DAN</td>
<td>VAN</td>
<td>FPN</td>
</tr>
</tbody>
</table>

**Within subnetworks**

<table>
<thead>
<tr>
<th>GroupINN</th>
<th>PCA</th>
<th>Diffpool</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRN</td>
<td>SM.M</td>
<td>SM.M</td>
</tr>
<tr>
<td>VAN</td>
<td>SM.H</td>
<td>SM.M</td>
</tr>
<tr>
<td>SN</td>
<td>AN</td>
<td>FPN</td>
</tr>
<tr>
<td>FPN</td>
<td>AN</td>
<td>MRN</td>
</tr>
<tr>
<td>SN</td>
<td>SM.H</td>
<td>SM.M</td>
</tr>
<tr>
<td>CON</td>
<td>SM.M</td>
<td>MRN</td>
</tr>
<tr>
<td>VAN</td>
<td>AN</td>
<td>SM.M</td>
</tr>
<tr>
<td>SM.M</td>
<td>CBLN</td>
<td>FPN</td>
</tr>
</tbody>
</table>

[Yujun Yan, Jiong Zhu, et al. ACM KDD ’19]

[Cohen, et al., J Neurosci 2016];
[Cole, et al. Neuron 2014];
[Davison, et al. PLOS Comp Bio 2015]
This talk: Summarization in Network Representation Learning

- Summarization within a GCN for faster training, data denoising and interpretability [ACM KDD’19a]

- Embedding summarization for compression and on-the-fly computation [ACM KDD’19b; PKDD’19]
Embeddings are powerful, but can take up a lot of space!

- For 1B nodes and K=128 → 1TB to store the embeddings!
- Can we summarize them?

[Di Jin, Rossi et al. ACM KDD’19]  
https://github.com/GemsLab/MultiLENS
Graph Summarization Survey

Network type
- Static
  - Plain
  - Labeled
- Dynamic
  - Plain

Structure only
- Grouping
- Compression
- Simplification
- Influence

Structure + labels
- Grouping
- Compression
- Influence

Temporal structure
- Grouping
- Compression
- Influence

Applications
- Query Efficiency
- Compression
- Visualization
- Pattern Discovery
- Influence Analysis

Open Problems
- Richer Data
- Streaming / Incremental Summarization
- Automated Insight Extraction
- Evaluation

Related Research Areas:
- Graph clustering, partitioning, community detection, sampling, sparsification, sketches, compression

Core techniques employed

[Cui, Li, Safavi, Dighe, Koutra. ACM Computing Surveys '18.]
Latent Network Summarization

• **Given:** a graph $G(V, E)$

• **Find:** a *compressed representation* that captures the key structural information and is

  - independent of graph size ($|V|$, $|E|$), and
  - capable of deriving node representations on the fly

[Di Jin, Rossi et al. ACM KDD’19]  
https://github.com/GemsLab/MultiLENS
## Comparison to Related Work

<table>
<thead>
<tr>
<th>Input</th>
<th>Input</th>
<th>Representations / Output</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation [2]</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Cosum [34]</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>AspEm [31]</td>
<td>✓</td>
<td>❌</td>
<td>✓</td>
</tr>
<tr>
<td>metapath2vec [8]</td>
<td>✓</td>
<td>❌</td>
<td>✓</td>
</tr>
<tr>
<td>n2vec [11], LINE [32]</td>
<td>❌</td>
<td>❌</td>
<td>✓</td>
</tr>
<tr>
<td>struc2vec [26]</td>
<td>❌</td>
<td>❌</td>
<td>✓</td>
</tr>
<tr>
<td>DNGR [6]</td>
<td>❌</td>
<td>❌</td>
<td>✓</td>
</tr>
<tr>
<td>GraphSAGE [12]</td>
<td>✓</td>
<td>❌</td>
<td>✓</td>
</tr>
<tr>
<td><strong>MULTI-LENS</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Latent Network Summarization: Overview

[Di Jin, Rossi et al. ACM KDD’19]  
https://github.com/GemsLab/MultiLENS
1. Relational functions to aggregate nodewise structural features automatically

[Di Jin, Rossi et al. ACM KDD’19]  https://github.com/GemsLab/MultiLENS
Latent Network Summarization: Overview

1. Relational functions to aggregate nodewise structural features automatically

2. Histogram-based heterogeneous contexts for nodes

[Di Jin, Rossi et al. ACM KDD’19]  https://github.com/GemsLab/MultiLENS
Latent Network Summarization: Overview

1. Relational functions to aggregate nodewise structural features automatically

2. Histogram-based heterogeneous contexts for nodes

3. Subspace vectors from which we can derive the embeddings

[Di Jin, Rossi et al. ACM KDD’19]  https://github.com/GemsLab/MultiLENS
Space comparison

<table>
<thead>
<tr>
<th>Data</th>
<th>SE</th>
<th>LINE</th>
<th>n2vec</th>
<th>DW</th>
<th>m2vec</th>
<th>AspEm</th>
<th>G2G</th>
<th>ML (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>facebook</td>
<td>8.13x</td>
<td>8.48x</td>
<td>12.79x</td>
<td>12.84x</td>
<td>3.82x</td>
<td>8.50x</td>
<td>9.17x</td>
<td>0.58</td>
</tr>
<tr>
<td>yahoo</td>
<td>187.1x</td>
<td>180.0x</td>
<td>242.2x</td>
<td>231.0x</td>
<td>79.8x</td>
<td>197.4x</td>
<td>195.8x</td>
<td>0.62</td>
</tr>
<tr>
<td>dbpedia</td>
<td>710.0x</td>
<td>714.2x</td>
<td>996.4x</td>
<td>996.2x</td>
<td>-</td>
<td>749.2x</td>
<td>743.6x</td>
<td>0.81</td>
</tr>
<tr>
<td>digg</td>
<td>608.2x</td>
<td>612.8x</td>
<td>848.9x</td>
<td>830.3x</td>
<td>259.9x</td>
<td>641.7x</td>
<td>635.2x</td>
<td>0.54</td>
</tr>
<tr>
<td>bibson.</td>
<td>1512.1x</td>
<td>1523.0x</td>
<td>2152.5x</td>
<td>2152.5x</td>
<td>-</td>
<td>1595.8x</td>
<td>-</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Multi-LENS requires 4-2152x less output storage space than the other embedding methods.

[Di Jin, Rossi et al. ACM KDD’19]  https://github.com/GemsLab/MultiLENS
The Multi-LENS node embeddings outperform all the baselines by 3.5–34.3% in AUC.
Inductive Anomaly Detection

- Learn summary of $G_{t-1}$, apply to $G_t$
- Compute the distance between the embeddings at $t-1$ and $t$

308K hashtags
2.6M edges
May -- July 2014

Gaza-Israel conflict
Ebola virus outbreak

https://github.com/GemsLab/MultiLENS
[Di Jin, Rossi et al. ACM KDD’19]
Can we summarize / compress the embeddings in a different way?
Temporal, Hash-based Node Embeddings

• **Given:** a time-evolving heterogeneous network $G(V, E)$

• **Learn:** a function $\chi: V \rightarrow \{0,1\}^d$ s.t. the derived $d$-dim embeddings
  1) preserve similarities in interactions in $G$,
  2) are space-efficient, and
  3) accurately capture temporal information and the heterogeneity of the underlying network

[Di Jin, Mark Heimann, et al. PKDD’19] [https://github.com/GemsLab/node2bits]
Example: Find similarities in user interactions

- **User stitching:**
  - The task of identifying and matching various online references to the *same user* in real-world web services.
  - Instance of *entity resolution*

[Cohen, W.W., Richman, J., KDD 2012]
[Dasgupta, A.+, WSDM 2012]
[Saha Roy, R.,+ WWW’15]
[Kim, Kini, + WWW’17]
[Bhattacharya, I., Getoor, L. TKDD 2007], …
node2bits: Key ideas

- [R1] Graph heterogeneity
  - General approach that aggregates rich features + node types

- [R2] Temporal dynamics
  - Temporally valid walks to capture short- and long-term interactions
  - Functionally similar nodes are represented by multiple features (structural sim)

- [R3] Efficient similarity comparison
  - Use LSH to hash similar nodes (linear complexity)

- [R4] Low storage requirement
  - Binary hashcode with fixed length

[D Ji, M Heimann, et al. PKDD’19] https://github.com/GemsLab/node2bits
node2bits: Workflow

[Di Jin, Mark Heumann, et al. PKDD’19]  https://github.com/GemsLab/node2bits
Q1: *Supervised* Identity stitching

<table>
<thead>
<tr>
<th>Metric</th>
<th>CN</th>
<th>SE</th>
<th>LINE</th>
<th>DW</th>
<th>n2vec</th>
<th>s2vec</th>
<th>DNGR</th>
<th>AspEm</th>
<th>CTDNE</th>
<th>N2B-0</th>
<th>N2B-SH</th>
<th>N2B-LN</th>
</tr>
</thead>
<tbody>
<tr>
<td>bitcoin</td>
<td>AUC</td>
<td>0.7474</td>
<td>0.5828</td>
<td>0.6071</td>
<td>0.6306</td>
<td>0.6462</td>
<td>0.8025</td>
<td>0.5909</td>
<td>0.5344</td>
<td>0.6987</td>
<td>0.7584</td>
<td>0.7609</td>
</tr>
<tr>
<td></td>
<td>ACC</td>
<td>0.7174</td>
<td>0.5842</td>
<td>0.5842</td>
<td>0.6158</td>
<td>0.6158</td>
<td>0.7263</td>
<td>0.5526</td>
<td>0.5316</td>
<td>0.6000</td>
<td>0.7211</td>
<td>0.7268</td>
</tr>
<tr>
<td></td>
<td>F1</td>
<td>0.7001</td>
<td>0.5728</td>
<td>0.5828</td>
<td>0.6158</td>
<td>0.6157</td>
<td>0.7263</td>
<td>0.5525</td>
<td>0.5315</td>
<td>0.5964</td>
<td>0.7209</td>
<td>0.7271</td>
</tr>
<tr>
<td>digg</td>
<td>AUC</td>
<td>0.6217</td>
<td>0.5171</td>
<td>0.7878</td>
<td>0.7398</td>
<td>0.7445</td>
<td>0.5105</td>
<td>0.6967</td>
<td>0.8185*</td>
<td>0.7611</td>
<td>0.7587</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACC</td>
<td>0.6217</td>
<td>0.5152</td>
<td>0.7694</td>
<td>0.6971</td>
<td>0.7013</td>
<td>OOT</td>
<td>OOM</td>
<td>0.7982*</td>
<td>0.7418</td>
<td>0.7444</td>
<td></td>
</tr>
<tr>
<td></td>
<td>F1</td>
<td>0.5585</td>
<td>0.3770</td>
<td>0.7683</td>
<td>0.6960</td>
<td>0.7003</td>
<td>0.5088</td>
<td>0.5884</td>
<td>0.7958*</td>
<td>0.7411</td>
<td>0.7433</td>
<td></td>
</tr>
<tr>
<td>wiki</td>
<td>AUC</td>
<td>0.6997</td>
<td>0.7854</td>
<td></td>
<td></td>
<td></td>
<td>0.5374</td>
<td>0.7707</td>
<td>0.8230</td>
<td>0.8259*</td>
<td>0.8214</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACC</td>
<td>0.6997</td>
<td>OOT</td>
<td>0.7132</td>
<td>OOM</td>
<td>OOM</td>
<td>OOT</td>
<td>OOM</td>
<td>0.7145</td>
<td>0.7510*</td>
<td>0.7103</td>
<td></td>
</tr>
<tr>
<td></td>
<td>F1</td>
<td>0.6699</td>
<td>0.7129</td>
<td></td>
<td></td>
<td></td>
<td>0.5141</td>
<td>0.6398</td>
<td>0.7088</td>
<td>0.7476*</td>
<td>0.7067</td>
<td></td>
</tr>
<tr>
<td>comp-X</td>
<td>AUC</td>
<td>0.5970</td>
<td>0.5000</td>
<td></td>
<td></td>
<td></td>
<td>0.5213</td>
<td></td>
<td>0.8095*</td>
<td>0.7496</td>
<td>0.7525</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACC</td>
<td>0.5970</td>
<td>OOM</td>
<td>0.6757</td>
<td>OOM</td>
<td>OOM</td>
<td>OOT</td>
<td>OOM</td>
<td>0.8414*</td>
<td>0.7959</td>
<td>0.7975</td>
<td></td>
</tr>
<tr>
<td></td>
<td>F1</td>
<td>0.5189</td>
<td>0.4032</td>
<td></td>
<td></td>
<td></td>
<td>0.5103</td>
<td></td>
<td>0.8154*</td>
<td>0.7581</td>
<td>0.7606</td>
<td></td>
</tr>
</tbody>
</table>

Dynamic + static variants of node2bits outperform baselines by up to 5.2% in AUC and 4.9% in F1 score. **Short-term** tactic performs better.

Q2: Output storage efficiency

node2bits uses 63-339× less space than the baselines, while achieving comparable or better stitching performance.

Summarizing Large Networks: Overview

“ranged star” attack

Structural Summaries
[SDM’14, KDD’15, Dat Bull Eng’17, SNAM’18, SDM’19, …]

Domain-specific Summaries
[ICDM’17, KDD’19]

Query-on-the-edge + Rule-based Summaries
[ICDM’19]

Interactive Summaries
[VLDB’15, Informatics’17]

Latent Summaries
[KDD’19]

Survey:
[CSUR’18]
Beyond Summarization

Structural embeddings for network alignment

When is it useful to learn over higher-order networks, and when is it not?

[Mark Heimann, Haoming Shen, Tara Safavi, Danai Koutra. ACM CIKM’18]

[Mark Heimann, Haoming Shen, Tara Safavi, Danai Koutra. ACM CIKM’18]

[Caleb Belth, Fahad Kamran, Donna Tjandra, Danai Koutra. IEEE/ACM ASONAM’19]

[Caleb Belth, Fahad Kamran, Donna Tjandra, Danai Koutra. IEEE/ACM ASONAM’19]
Take-away messages: Summarization in Network Representation Learning

- Structural embeddings are less studied, but are more appropriate than proximity-based ones in several tasks.
- Summarization within a GCN can help with faster training, data denoising and interpretability [ACM KDD’19a].
- Embedding summarization can achieve compression and on-the-fly computation of representations [ACM KDD’19b; PKDD’19].
- Histograms are powerful at capturing the graph structure [ACM CIKM’18; ACM KDD’19b,c; ECML/PKDD’19; IEEE ICDM’19].
  - flexible, versatile (heterogeneity, attributes, directionality, weights…),
  - less information loss.
Talk based on the following papers

Thank you!

Questions?

http://danaikoutra.com
dkoutra@umich.edu

https://github.com/GemsLab/GroupINN
https://github.com/GemsLab/MultiLENS
https://github.com/GemsLab/node2bits