Frontiers of Graph Neural Networks with DIG

https://github.com/divelab/DIG/

Shuiwang Ji, Yaochen Xie, Zhao Xu, Haiyang Yu
<table>
<thead>
<tr>
<th>Session</th>
<th>Duration</th>
<th>Presenter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph neural networks</td>
<td>20 min</td>
<td>Shuiwang Ji</td>
</tr>
<tr>
<td>Self-supervised learning</td>
<td>35 min</td>
<td>Yaochen Xie</td>
</tr>
<tr>
<td>Practical session: <em>dig.sslgraph</em></td>
<td>15 min</td>
<td>Zhao Xu</td>
</tr>
<tr>
<td>Q&amp;A</td>
<td>10 min</td>
<td>All</td>
</tr>
<tr>
<td>Explainability</td>
<td>35 min</td>
<td>Haiyang Yu</td>
</tr>
<tr>
<td>Practical session: <em>dig.xgraph</em></td>
<td>15 min</td>
<td>Haiyang Yu</td>
</tr>
<tr>
<td>Q&amp;A</td>
<td>10 min</td>
<td>All</td>
</tr>
</tbody>
</table>
Success of Deep Learning

Deep learning shines on many tasks

Image Classification

Machine Translation

Speech recognition
Objects + Relationships = Graph
Tasks over Graphs

- Node-level classification/regression
- Graph-level classification/regression
- Link prediction
- Generation
- Explainability
- ......
Tasks on Graph Data

• e.g., Node classification
  – Document classification in a citation network
Tasks on Graph Data

- e.g., Graph classification
  - Molecular property prediction

Toxic

Non-toxic

?
Tasks on Graph Data

• e.g., Link prediction
  – Friend suggestion in a social network
Graph Neural Networks: Basic Concepts

https://github.com/divelab/DIG/
Notations

- $G = (X, A)$
- Adjacency matrix
  
  $A: \begin{bmatrix}
  0 & 1 & 1 & 1 \\
  1 & 0 & 0 & 0 \\
  1 & 0 & 0 & 1 \\
  1 & 0 & 1 & 0 \\
  \end{bmatrix} \in \mathbb{R}^{N \times N}$
- Feature matrix
  
  $X: \begin{bmatrix}
  9 & 2 \\
  5 & 3 \\
  4 & 4 \\
  1 & 6 \\
  \end{bmatrix} \in \mathbb{R}^{N \times d}$
  
- We will consider edge features later
Permutation

\[
X: \begin{bmatrix} 9 & 2 \\ 5 & 3 \\ 4 & 4 \\ 1 & 6 \end{bmatrix} \in \mathbb{R}^{N \times d} \quad \quad A:\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{N \times N}
\]
Permutation

\[ X: \begin{bmatrix} 9 & 2 \\ 5 & 3 \\ 4 & 4 \\ 1 & 6 \end{bmatrix} \in R^{4 \times d} \]

\[ A: \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \in R^{4 \times 4} \]

\[ X': \begin{bmatrix} 5 & 3 \\ 1 & 6 \\ 9 & 2 \\ 4 & 4 \end{bmatrix} \in R^{4 \times d} \]

\[ A': \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \in R^{4 \times 4} \]
Permutation

\[
X : \begin{bmatrix}
9 & 2 \\
5 & 3 \\
4 & 4 \\
1 & 6
\end{bmatrix} \in \mathbb{R}^{N \times d}
\]

\[
A : \begin{bmatrix}
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0
\end{bmatrix} \in \mathbb{R}^{N \times N}
\]

Permutation matrix:

\[
P : \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix} \in \{0,1\}^{N \times N}
\]

\[
X' = PX; A' = PAP^T
\]

\[
X' : \begin{bmatrix}
5 & 3 \\
1 & 6 \\
9 & 2 \\
4 & 4
\end{bmatrix} \in \mathbb{R}^{N \times d}
\]

\[
A' : \begin{bmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
0 & 1 & 1 & 0
\end{bmatrix} \in \mathbb{R}^{N \times N}
\]
For graph-level representations

- We learn a function $f$ that maps a graph $G = (X, A)$ to a representation vector $\in \mathbb{R}^{d'}$

$$f(X, A) = f(PX, PAP^T)$$

- $P$ is any permutation matrix
Permutation Equivariance

• For node-level representations
  – We learn a function $f$ that maps a graph $G = (X, A)$ to a node representation matrix $\mathbf{E} \in \mathbb{R}^{N \times d'}$

\[
P f(X, A) = f(PX, PAP^T)
\]
Graph Neural Networks

Partially based on materials at:
https://geometricdeeplearning.com/lectures/
Overview

- GNNs are composed of multiple permutation equivariant/invariant layers/functions.

• **Neighborhood aggregation!**
  
  - Neighborhood features $X_{N_i} = \{x_j : j \in N_i\}$
  - Define a local function $\phi(x_i, X_{N_i})$

\[
H = f(X, A) = \begin{bmatrix}
\phi(x_1, X_{N_1}) \\
\vdots \\
\phi(x_N, X_{N_N})
\end{bmatrix}
\]
Blueprint for Learning on Graphs

Department of Computer Science & Engineering
Blueprint for Learning on Graphs


Department of Computer Science & Engineering
Blueprint for Learning on Graphs

Blueprint for Learning on Graphs


Department of Computer Science & Engineering
What’s in a GNN layer

- Neighborhood aggregation!
  - We build permutation equivalent functions $f(X, A)$ on graphs by sharing the local permutation invariant function $\phi(x_i, x_{N_i})$
  - How to implement $\phi$?

\[
H = f(X, A) = \begin{bmatrix}
\phi(x_1, x_{N_1}) \\
\vdots \\
\phi(x_N, x_{N_N})
\end{bmatrix}
\]
Three “flavours” of GNN layers

**Convolutional**

\[ h_u = \phi \left( x_u, \bigoplus_{v \in N_u} c_{uv} \psi(x_v) \right) \]

**Attentional**

\[ h_u = \phi \left( x_u, \bigoplus_{v \in N_u} a(x_u, x_v) \psi(x_v) \right) \]

**Message-passing**

\[ h_u = \phi \left( x_u, \bigoplus_{v \in N_u} \psi(x_u, x_v) \right) \]


Department of Computer Science & Engineering
Frontier Topics of GNN

- **Self-supervised learning of GNNs**
  - Studies how to train GNNs with unlabeled graph data
  - Applications: pre-training in drug discovery, node representation learning for industrial large-scale graphs.

- **GNN explainability**
  - Studies the cause of GNN predictions
  - Applications: building trustworthy and transparent GNN models

- **Graph generation, 3D geometric GNNs, etc.**
Self-Supervised Learning on Graphs

https://github.com/divelab/DIG/
Self-supervised Learning

- Training models with self-supervision.
- Success in text and image data. We see an explosion of graph SSL papers.

- Three paradigms of using self-supervised learning
  - Unsupervised representation learning
  - Unsupervised pre-training
  - Auxiliary learning

Xie et al. Self-Supervised Learning of Graph Neural Networks: A Unified Review. TPAMI, 2022.
Contrastive vs Predictive: are negative pairs required?

- **Contrastive**
  - Maximizes mutual information
  - Focuses on designing view generation

- **Predictive**
  - Ad-hoc pretext-tasks / information bottleneck
  - Focuses on designing task/objective

Xie et al. Self-Supervised Learning of Graph Neural Networks: A Unified Review. TPAMI, 2022.
Taxonomy of SSL Methods

Xie et al. Self-Supervised Learning of Graph Neural Networks: A Unified Review. TPAMI, 2022.
Contrastive Methods

- Representations of “similar” graph data to be closer, and that of “dissimilar” graph data to be further from each other.

- In practice, we do not have ground-truth or defined measurement for similarity. We construct augmented graphs (or subgraphs) from the same graph to be “similar”, and different graph samples are “dissimilar”.
Contrastive Methods

- Representations of “similar” graph data to be closer, and that of “dissimilar” graph data to be further from each other.

- Theoretical grounding: mutual information maximization. Jointly sampled graphs are “similar”, while independently sampled graphs are “dissimilar”

\[ I(x, y) = D_{KL}(p(x, y) \| p(x)p(y)) \]

- We consider a graph as a random variable and obtain two views of the graph.
- Representations of the two views are x and y. A good GNN should encode two views into representation that share as much mutual information as possible.
Contrastive Methods

View generation

Input data → View 1
Input data → View 2

Encoding views

Encoder

Objectives

Representation 1

$MI$ maximization

Representation 2

Xie et al. Self-Supervised Learning of Graph Neural Networks: A Unified Review. TPAMI, 2022.
Contrastive Objectives

- Derived as lower bounds of mutual information
  - Jensen-Shannon Estimator (DGI, InfoGraph, MVGRL, etc.)
    \[
    \hat{I}^{(JS)}(h_i, h_j) = \mathbb{E}_{(A, X) \sim P}[\log(D(h_i, h_j))] + \mathbb{E}_{(A, X), (A', X') \sim P \times P}[\log(1 - D(h_i, h'_j))]
    \]
  - InfoNCE (GraphCL, GRACE, etc.)
    \[
    \hat{I}^{(NCE)}(h_i, h_j) = \mathbb{E}_{(A, X) \sim P}[D(h_i, h_j)] - \mathbb{E}_{K \sim P^N}\left[\log \sum_{(A', X') \in K} e^{D(h_i, h_j')}/N \right](A, X)
    \]
    \[
    = \mathbb{E}_{(A, X), K \sim P \times P^N}\left[\log \frac{e^{D(h_i, h_j)}}{\mathbb{E}_{(A', X') \in K} e^{D(h_i, h'_j)}}\right] + \log N
    \]

Xie et al. Self-Supervised Learning of Graph Neural Networks: A Unified Review. TPAMI, 2022.
• Three types of graph transformations for view generation

Xie et al. Self-Supervised Learning of Graph Neural Networks: A Unified Review. TPAMI, 2022.
What are good views for graphs?

- Ideally, good views \((v_1, v_2)\) generated for contrastive learning should
  - Have their mutual information \(I(v_1, v_2)\) minimized
  - Subject to \(I(v_1, y) = I(v_2, y) = I(x, y)\)

- The principle is even more important for graphs.

BGRL

\[ \theta \leftarrow \text{optimize}(\theta, \eta, \partial_\theta \ell(\theta, \phi)) \quad \text{EMA update: } \phi \leftarrow \tau \phi + (1 - \tau) \theta \]

Thakoor et al., Large-Scale Representation Learning on Graphs via Bootstrapping. ICLR 2022.
CCA-SSG

\[ G = (X, A) \]

1. random augmentation

2. graph neural network

\[ GNN_\theta \]

3. CCA-based objective

\[ CCA \]

\[ \mathcal{L} = \left\| \tilde{Z}_A - \tilde{Z}_B \right\|_F^2 + \lambda \left( \left\| \tilde{Z}_A^\top \tilde{Z}_A - I \right\|_F^2 + \left\| \tilde{Z}_B^\top \tilde{Z}_B - I \right\|_F^2 \right) \]

- invariance term
- decorrelation term

Zhang et al., From Canonical Correlation Analysis to Self-supervised Graph Neural Networks. NeurIPS 2021.
Predictive - Invariance Regularization

- LaGraph

\[ L_{\text{node}}(\mathcal{E}, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} \| \mathcal{D}(A_i, H_i) - X_i \|^2 + \alpha \left[ \frac{\sum_i \| 1_{J_i} \odot H_i - 1_{J_i} \odot H'_i \|^2}{\sum_i |J_i|} \right]^{1/2} + \alpha' \left[ \frac{\sum_i \| z_i - z'_i \|^2 / \sum_i |J_i|}{\sum_i |J_i|} \right]^{1/2} \]

Xie et al., Self-Supervised Representation Learning via Latent Graph Prediction. ICML 2022.
Predictive - Invariance Regularization

● Similarity
  ○ Do not requires negative samples.
  ○ Better scalability.
  ○ Their objectives include a term to regularize the invariance of representations to different views (minimize their distance)
  ○ Effectiveness can be justified by the Information Bottleneck Principle

● Difference
  ○ The formulation of entire objectives are different.
  ○ Based on or derived from different theoretical grounding/insights
  ○ Different technique to guarantee informative representations
Next session: Explainability of GNNs
Explainability of GNNs

https://github.com/divelab/DIG/
Why Explainability

Good performance but

- People don’t understand deep model
  - taking it as black box
- People don’t trust deep model
  - when the model fails

Explainability:

- Provide human-intelligible explanations to build trustworthy AI

Reference: https://towardsdatascience.com/guide-to-interpretable-machine-learning-d40e8a64b6cf
Challenges

● How to explain GNNs with edge information
  ○ Instance-level: edge-based, subgraph-based
  ○ Model-level: xgnn

● How to get human-understandable explanation results
  ○ Graph Sentiment dataset

● How to compare explainability methods systematically
  ○ Quantitative metrics: Fidelity+, Fidelity-, Sparsity

Yuan et al. Explainability in Graph Neural Networks: A Taxonomic Survey. TPAMI 2022.
Overview of methods

Yuan et al. Explainability in Graph Neural Networks: A Taxonomic Survey. TPAMI, 2022
Explanation on GNNs

Assigning importance scores on edges

Assigning importance scores on connected subgraphs

Importance score for each subgraph:
- 0.5
- 0.7
- 0.5
- 0.4
A: Edge masks for class-distinguish important edges
B: Node masks for important node feature dimension.

Explanation based on edges

\[
\min_M - \sum_{c=1}^C 1[y = c] \log P_\Phi(Y = y \mid G = A_c \odot \sigma(M_E), X = X_c \odot \sigma(M_X))
\]

- It learns soft masks on graph edges and node features.
- The masks are randomly initialized and updated to maximum the mutual information between the original predictions and the perturbed graphs.

Ying et al. GNNEexplainer: Generating Explanations for Graph Neural Networks. NeurIPS, 2019.
SubgraphX

Explanation based on subgraphs

\[ G^* = \underset{|G_i| \leq N_{\text{min}}}{\text{argmax}} \text{Score}(f(\cdot), G, G_i) \]

- Explore the important subgraphs with Monte Carlo tree search algorithm

\[ a^* = \underset{a_j}{\text{argmax}} Q(N_i, a_j) + U(N_i, a_j) \]

- Take Shapley value as the subgraph importance scores

\[ \varphi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n-|S|-1)!}{n!} (v(S \cup \{i\}) - v(S)) \]

_Yuan et al._ On Explainability of Graph Neural Networks via Subgraph Explorations. ICML 2021.
SubgraphX

Yuan et al. On Explainability of Graph Neural Networks via Subgraph Explorations. ICML 2021.
Better human evaluations for the explainability results

- Graph-SST2, Graph-SST5, Graph-Twitter

Yuan et al. Explainability in Graph Neural Networks: A Taxonomic Survey. TPAMI 2022.
Yuan et al. On Explainability of Graph Neural Networks via Subgraph Explorations. ICML 2021.
Evaluation Metrics

- Compare different explainability methods systematically
  - Fidelity+, Fidelity-, Sparisty

Benchmark of explainability methods using Fidelity and Sparsity

Yuan et al. Explainability in Graph Neural Networks: A Taxonomic Survey. TPAMI 2022.
Thank you!

Q&A

DIG

xgraph
benchmark

Dive into Graphs
Q&A