Graph Representation Learning with Hierarchical Structure and Domain Adaptation

Speaker: Lun Du
Affiliation: Microsoft Research Asia
Main Contents

- Background
- Graph embedding with hierarchical community structure
- Domain adaptive graph embedding
- Future works
Background

- **Graph Embedding** tries to map graph vertices into a low-dimensional vector space under the condition of preserving different types of graph properties.

- Node classification
- Link Prediction
- Network Visualization
- Community detection
- ……
Background

- Unsupervised vs. Supervised
  - DeepWalk, LINE, node2vec, etc.
  - GCN, GraphSAGE, etc.

- Euclidean vs. Non-Euclidean
  - Hyperbolic space (Tag2Vec, WWW’19)

- Vector vs. Distribution
  - Using variance to model uncertainty of semantic
Main Contents

- Background
- Graph embedding with hierarchical community structure
- Domain adaptive graph embedding
- Future works
Conceptually, complex networks have **hierarchical community** in real world.

- E.g. social networks, air transportation networks, and metabolic networks, etc.
Hierarchical Info can be observed to a certain extent in online networks.

Explicit hierarchy with attributes

Implicit hierarchy with tags

Facebook Network

Twitter Network
Goal:
- Encoding the rich hierarchical structural information

Main Challenges:
- How to represent nodes or tags?
- How to learn the representations effectively and efficiently?
Galaxy Network Embedding:
A Hierarchical Community Structure Preserving Approach

How to Represent?

- Inspired by galaxy structure
- Embedding nodes and communities simultaneously
  - Easy to analyze the network at different scales.

(The representations of nodes in tree)
How to Learn?

- Formulate the **hierarchical community preserving** network embedding
  - One is the local information, i.e. pairwise nodes similarity in the same community.
  - The other is the hierarchical structure property, i.e. horizontal relationship and vertical relationship.
- Implement and optimize efficiently the embedding method.
Hierarchical Preserving Network Embedding

- Pairwise Proximity Preservation

\[
\min_{\Phi, \Phi'} O_{k}^{(l-1)} = - \sum_{c_i^l, c_j^l \in Ch(c_{k}^{l-1})} S_{i,j}^l \log P(\Phi'(c_j^l) | \Phi(c_i^l))
\]

\[
S_{i,j}^l = \frac{1}{|c_i^l||c_j^l|} \sum_{u \in c_i^l} \sum_{v \in c_j^l} \frac{A_u^T A_v}{\sqrt{||A_u||_1 ||A_v||_1}}
\]

\[
P(\Phi'(c_j^l) | \Phi(c_i^l)) = \frac{\exp(\Phi'(c_j^l) \cdot \Phi(c_i^l))}{\sum_{c_i^l \in Ch(c_{k}^{l-1})} \exp(\Phi'(c_i^l) \cdot \Phi(c_i^l))},
\]

2019/10/11

Graph Representation Learning
Hierarchical Preserving Network Embedding

- **Hierarchical Structure Preservation**
  - Horizontal relationship:
    \[ \| \Phi(c_u^l) - \Phi(c_v^l) \| < \| \Phi(c_u^l) - \Phi(c_w^l) \|, \quad (3) \]
  - Vertical relationship:
    \[ \| \Phi(c_{i+1}^l) - \Phi(c_j^l) \| < \| \Phi(c_j^l) - \Phi(c_{k-1}^l) \|. \quad (4) \]
Galaxy Network Embedding

Objective

\[
\min_{\Phi, \Phi'} O_k^{(l-1)} = - \sum_{c_i^l, c_j^l \in Ch(c_k^{l-1})} S_{i,j}^l \log P(\Phi'(c_j^l) \mid \Phi(c_i^l))
\]

s.t. \( \forall c_i^l \in Ch(c_k^{l-1}), \| \Phi(c_i^l) - \Phi(c_k^{l-1}) \|_2 = r_k^{l-1} \).

where,

\[
r_i^l = \eta \cdot d_k^{l-1}, \quad \eta < \frac{1}{6}
\]

\[
d_k^{l-1} = \min_{c_i^l, c_j^l \in Ch(c_k^{l-1}), i \neq j} \text{Dist} (\Phi(c_i^l), \Phi(c_j^l)),
\]

\[
\text{Dist}(x, y) = \| x - y \|,
\]

Figure 2: Structure of GNE
Proof

Galaxy Network Embedding

\[
\min_{\Phi, \Phi'} O_k^{(l-1)} = - \sum_{c_i', c_j' \in Ch(c_{k}^{-1})} S_{i,j}^{l} \log P(\Phi'(c_j') | \Phi(c_i')) \\
\text{s.t.} \quad \forall c_i' \in Ch(c_{k}^{-1}), \quad \|\Phi(c_i') - \Phi(c_{k}^{-1})\|_2 = r_k^{l-1}. \tag{5}
\]

where,

\[
r_i^l = \eta \cdot d_{k}^{l-1}, \quad \eta < \frac{1}{6}
\]

\[
d_k^{l-1} = \min_{c_i', c_j' \in Ch(c_{k}^{-1}), i \neq j} \text{Dist} (\Phi(c_i'), \Phi(c_j')) \tag{6}
\]

\[
\text{Dist}(x, y) = \|x - y\|
\]

Hierarchical Preserving Network Embedding

- **Pairwise Proximity Preservation**

\[
\min_{\Phi, \Phi'} O_k^{(l-1)} = - \sum_{c_i', c_j' \in Ch(c_{k}^{-1})} S_{i,j}^{l} \log P(\Phi'(c_j') | \Phi(c_i'))
\]

- **Hierarchical Structure Preservation**

  - **Horizontal relationship:**
    \[
    \|\Phi(c_{u}^{'}) - \Phi(c_w^{'})\| < \|\Phi(c_{u}^{'}) - \Phi(c_{w}^{'})\|, \tag{3}
    \]
  
  - **Vertical relationship:**
    \[
    \|\Phi(c_{i}^{l+1}) - \Phi(c_j^{'})\| < \|\Phi(c_j^{'}) - \Phi(c_{k}^{l-1})\|. \tag{4}
    \]
Proof

**Lemma 1**

The community representations learned from recursively optimizing the objective Eq.(5) with the strategy Eq.(6) preserve the constraints Eq.(3) and Eq.(4).

\[
\begin{align*}
    r_i^l &= \eta \cdot d_{k}^{l-1}, \quad \eta < \frac{1}{6} \\
    d_k^{l-1} &= \min_{c_i^l, c_j^l \in Ch(c_{k}^{l-1}), i \neq j} \text{Dist}(\Phi(c_i^l), \Phi(c_j^l)) \\
    \text{Dist}(x, y) &= \|x - y\|,
\end{align*}
\]

\[\Rightarrow\]

**Horizontal relationship:**

\[\|\Phi(c_u^l) - \Phi(c_v^l)\| < \|\Phi(c_u^l) - \Phi(c_w^l)\|,\]

**Vertical relationship:**

\[\|\Phi(c_i^{l+1}) - \Phi(c_j^l)\| < \|\Phi(c_j^l) - \Phi(c_{k}^{l-1})\|.\]
Experiment

Dataset

- Facebook social network datasets:
  - Amherst College
  - Hamilton University
  - Georgetown University
- Hierarchical random graphs (HRG):
  - syn_with_125nodes
  - syn_with_1800nodes
  - syn_with_2560nodes
  - syn_with_3750nodes

Baselines

- Spectral Clustering [Tang and Liu, 2011]
- DeepWalk [Perozzi et al, 2014]
- Node2vec [Grover and Leskovec, 2016]
- LINE [Tang et al., 2017]
- GraRep [Cao et al., 2015]
- MNMF [Wang et al., 2017]
Hierarchical Community Detection

Figure 3: The comparison of hierarchical community preservation on different models. Three different structures of HRG with the same number of layers are used.
Network Visualization

Figure 4: The visualization of vertex representations on different models
## Vertex Classification

<table>
<thead>
<tr>
<th>Model</th>
<th>Amherst</th>
<th>Hamilton</th>
<th>Georgetown</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10%</td>
<td>30%</td>
<td>50%</td>
</tr>
<tr>
<td>GNE</td>
<td>93.57</td>
<td>93.31</td>
<td><strong>93.33</strong></td>
</tr>
<tr>
<td>SpectralClustering</td>
<td>72.89</td>
<td>73.49</td>
<td>73.94</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>90.62</td>
<td>91.65</td>
<td>91.32</td>
</tr>
<tr>
<td>Node2Vec</td>
<td>91.29</td>
<td>91.24</td>
<td>91.04</td>
</tr>
<tr>
<td>LINE</td>
<td>90.76</td>
<td>91.82</td>
<td>91.48</td>
</tr>
<tr>
<td>GraRep</td>
<td>92.13</td>
<td>92.25</td>
<td>91.78</td>
</tr>
<tr>
<td>MNMF</td>
<td>89.82</td>
<td>89.06</td>
<td>88.04</td>
</tr>
</tbody>
</table>

Table 1: The multi-label classification results on different percentages of test datasets
Hierarchical Community Structure Preserving Network Embedding: A Subspace Approach

Drawbacks of GNE

- Failed when embedding deeper communities
  - Radii shrink exponentially
  - Data sparsity in a deeper community
- Probably overvalued hierarchical information
  - Vertices across community are exponentially distant than those within the same community.

Figure 5: The comparison of hierarchical community preservation on different models. A 6-layer generated hierarchical networks is used.
How to Represent?

- Subspace
  - Natural hierarchical structure
  - Deeper community corresponding to lower dimensional subspace

Figure 1: The correspondence between the community hierarchy and the subspace hierarchy
How to Learn?

- Formulating the problem into an optimization problem with subspace constraints
  - Modeling community affiliation by subspace
  - Reducing the representation dim by constraining the rank of base vectors
- Designing efficient learning algorithm
  - From global to layer-wise optimization
  - From discrete to differentiable optimization
Hierarchical Structure Preserved

- Preservation of Structure within Individual Communities
  \[ \mathcal{L}_1 = \sum_{(i,j) \in E} \log \sigma(||\overline{u}_j^{(0)} - \overline{u}_i^{(0)}||) + k \cdot \mathbb{E}_{v_n \sim p_n} [\log \sigma(-||\overline{u}_n^{(0)} - \overline{u}_i^{(0)}||)] \]
  Where, \( \overline{u}_i^{(l)} = S_{f_i} \overline{u}_i^{(l-1)} \) for \( l = 1 \ldots L \), \( v_i \in V \)

- Preservation of Structure among Communities
  \[ \mathcal{L}_2 = \sum_{l=0}^{L} \sum_{i=1}^{\left| C_i \right|} \sum_{j=i+1}^{\left| C_i \right|} (\Delta_{i,j} - \Gamma_{i,j})^2 \]

- Low Rank Constraints
  \[ \mathcal{L}_3 = \sum_{l=0}^{L} \sum_{i=1}^{\left| C_i \right|} \text{rank}(S^l_i) \]
**Vertex Classification**

<table>
<thead>
<tr>
<th>Model</th>
<th>Amherst</th>
<th></th>
<th></th>
<th></th>
<th>Georgetown</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>UC</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30%</td>
<td>50%</td>
<td>70%</td>
<td>90%</td>
<td>30%</td>
<td>50%</td>
<td>70%</td>
<td>90%</td>
<td>30%</td>
<td>50%</td>
<td>70%</td>
<td>90%</td>
<td>30%</td>
</tr>
<tr>
<td>SpaceNE</td>
<td>92.52</td>
<td>93.11</td>
<td>93.74</td>
<td>95.09</td>
<td>56.12</td>
<td>56.42</td>
<td>56.92</td>
<td>56.54</td>
<td>88.69</td>
<td>89.02</td>
<td>89.23</td>
<td>90.07</td>
<td></td>
</tr>
<tr>
<td>GNE</td>
<td>93.17</td>
<td>93.33</td>
<td>93.26</td>
<td>93.52</td>
<td>52.19</td>
<td>53.53</td>
<td>53.75</td>
<td>53.12</td>
<td>87.78</td>
<td>88.42</td>
<td>88.42</td>
<td>87.57</td>
<td></td>
</tr>
<tr>
<td>MNMF</td>
<td>87.11</td>
<td>88.04</td>
<td>89.23</td>
<td>89.96</td>
<td>51.52</td>
<td>51.69</td>
<td>51.60</td>
<td>53.25</td>
<td>87.89</td>
<td>87.95</td>
<td>88.09</td>
<td>88.10</td>
<td></td>
</tr>
<tr>
<td>DeepWalk</td>
<td>91.09</td>
<td>91.26</td>
<td>91.71</td>
<td>92.03</td>
<td>51.45</td>
<td>53.25</td>
<td>53.76</td>
<td>54.03</td>
<td>88.35</td>
<td>88.42</td>
<td>88.51</td>
<td>88.63</td>
<td></td>
</tr>
<tr>
<td>LINE</td>
<td>91.11</td>
<td>91.53</td>
<td>91.89</td>
<td>91.67</td>
<td>51.35</td>
<td>51.93</td>
<td>52.18</td>
<td>52.38</td>
<td>87.71</td>
<td>87.88</td>
<td>87.95</td>
<td>87.53</td>
<td></td>
</tr>
<tr>
<td>Struc2Vec</td>
<td>72.72</td>
<td>73.35</td>
<td>73.92</td>
<td>77.23</td>
<td>46.85</td>
<td>47.44</td>
<td>48.33</td>
<td>47.59</td>
<td>87.96</td>
<td>87.89</td>
<td>88.11</td>
<td>88.25</td>
<td></td>
</tr>
<tr>
<td>SpectralClustering</td>
<td>72.88</td>
<td>73.51</td>
<td>73.89</td>
<td>74.41</td>
<td>49.67</td>
<td>50.02</td>
<td>50.79</td>
<td>51.23</td>
<td>84.23</td>
<td>84.35</td>
<td>84.31</td>
<td>84.21</td>
<td></td>
</tr>
</tbody>
</table>

*Table 2: The multi-label classification results on different percentages of train datasets*
## Link Prediction

<table>
<thead>
<tr>
<th>Model</th>
<th>Amherst</th>
<th>Georgetown</th>
<th>UC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SpaceNE</td>
<td>85.61</td>
<td>89.28</td>
<td>91.32</td>
</tr>
<tr>
<td>GNE</td>
<td>62.07</td>
<td>68.97</td>
<td>51.25</td>
</tr>
<tr>
<td>MNMF</td>
<td>48.89</td>
<td>49.76</td>
<td>50.05</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>86.40</td>
<td>89.16</td>
<td>91.39</td>
</tr>
<tr>
<td>LINE</td>
<td>74.37</td>
<td>76.58</td>
<td>71.22</td>
</tr>
<tr>
<td>Struc2Vec</td>
<td>51.77</td>
<td>49.94</td>
<td>46.83</td>
</tr>
<tr>
<td>SpectralClustering</td>
<td>37.76</td>
<td>40.63</td>
<td>38.68</td>
</tr>
</tbody>
</table>

*Table 4: The link prediction results on different datasets.*
Experiment

Network Visualization

(a) SpaceNE  (b) GNE  (c) MNMF
(d) Stru2vec  (e) LINE  (f) DeepWalk
Hierarchical Info can be observed to a certain extent in online networks.

Explicit hierarchy with attributes

Implicit hierarchy with tags

Facebook Network

Twitter Network
Tag2Gauss: Learning Tag Representations via Gaussian Distribution in Tagged Networks

How to Represent?

- Represent tags and nodes simultaneously
- Tags represent node communities with intricate overlapping relationships
- Distribution: Tag; Sample from distributions: Node
How to learn?

Tag2Gauss Framework:

- Tag-view Embedding
- Node-view Embedding
- Multi-task Learning
Experiments

- **Datasets:**
  - Leetcode (652 nodes, 1096 edges, 34 tags, 3 labels)
  - Bilibili (11727 nodes, 187148 edges, 151 tags, 10 labels)
  - Cora. (2707 nodes, 5429 edges, 1433 tags, 7 labels)

- **Baselines**
  - DeepWalk (KDD’14)
  - Node2vec (KDD’16)
  - Hybrid Deepwalk (Naive Design)
  - GraphSage (NIPS’17)
The Advantage of Distribution Representations
# Node Classification

<table>
<thead>
<tr>
<th>Model</th>
<th>Leetcode</th>
<th></th>
<th></th>
<th></th>
<th>Bilibili</th>
<th></th>
<th></th>
<th></th>
<th>Cora</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10%</td>
<td>30%</td>
<td>50%</td>
<td>70%</td>
<td>90%</td>
<td>10%</td>
<td>30%</td>
<td>50%</td>
<td>70%</td>
<td>90%</td>
<td>10%</td>
<td>30%</td>
</tr>
<tr>
<td>Node2Vec</td>
<td>36.37%</td>
<td>36.37%</td>
<td>38.68%</td>
<td>37.63%</td>
<td>39.68%</td>
<td>48.19%</td>
<td>48.19%</td>
<td>45.36%</td>
<td>45.36%</td>
<td>42.88%</td>
<td>57.12%</td>
<td>57.40%</td>
</tr>
<tr>
<td>LINE</td>
<td>34.41%</td>
<td>38.59%</td>
<td>35.89%</td>
<td>33.66%</td>
<td>40.46%</td>
<td>6.55%</td>
<td>7.21%</td>
<td>7.65%</td>
<td>8.30%</td>
<td>9.28%</td>
<td>49.00%</td>
<td>49.96%</td>
</tr>
<tr>
<td>GraphSage</td>
<td>34.00%</td>
<td>37.37%</td>
<td>36.65%</td>
<td>39.77%</td>
<td>44.37%</td>
<td>61.48%</td>
<td>60.81%</td>
<td>60.52%</td>
<td>59.02%</td>
<td>54.26%</td>
<td>50.95%</td>
<td>51.63%</td>
</tr>
<tr>
<td>Tag2Gauss</td>
<td><strong>42.27%</strong></td>
<td><strong>42.68%</strong></td>
<td><strong>43.70%</strong></td>
<td><strong>44.04%</strong></td>
<td><strong>45.03%</strong></td>
<td><strong>61.65%</strong></td>
<td><strong>61.23%</strong></td>
<td><strong>60.83%</strong></td>
<td><strong>60.58%</strong></td>
<td><strong>56.85%</strong></td>
<td><strong>68.45%</strong></td>
<td><strong>67.21%</strong></td>
</tr>
</tbody>
</table>

Table 1: The comparison of node classification measured by Macro-$F_1$ on different models and different training size.
Main Contents

- Background
- Graph embedding with hierarchical community structure
- Domain adaptive graph embedding
- Future works
DANE: Domain Adaptive Network Embedding

Motivation

- Domain adaptation
  - Transferring machine learning models across different datasets to handle the same task
- Domain adaptation on networks is significant:
  - Reduce the cost of training downstream machine learning models by enabling models to be reused on other networks
  - Handle the scarcity of labeled data by transferring models trained well on a labeled network to unlabeled networks
- It is important to design a network embedding algorithm that can support domain adaptation.
Challenges

- Embedding space alignment
  - Structurally similar nodes should have similar representations in the embedding space, even if they are from different networks.

- Distribution alignment
  - Embedding vectors of different networks should have similar distribution in embedding space.
  - Most machine learning models perform as guaranteed only when they work on data with similar distribution as their training data.
Technique Framework: Overall

Figure 1: An overview of DANE. DANE consists of two major components: (a) shared weight graph convolutional network (SWGCN) projects the nodes from two networks into a shared embedding space and preserve cross-network similarity; (b) adversarial learning regularization is a two-player game where the first player is a discriminator trained to distinguish which network a representation vector is from and the second player is the SWGCN trying to generate embeddings that can confuse the discriminator.
Adversarial Learning Regularization

- Discriminator to avoid the instability of adversarial learning:

\[ L_D = \mathbb{E}_{x \in V_{src}} [(D(x) - 0)^2] + \mathbb{E}_{x \in V_{tgt}} [(D(x) - 1)^2] \]

- Adversarial training loss function to confuse the discriminator is:

\[ L_{adv} = \mathbb{E}_{x \in V_{src}} [(D(x) - 1)^2] + \mathbb{E}_{x \in V_{tgt}} [(D(x) - 0)^2] \]

- Overall loss function

\[ L = L_{gcn} + \lambda L_{adv} \]
## Experiment

### Comparison with Baselines

<table>
<thead>
<tr>
<th>Methods</th>
<th>Paper Citation Network</th>
<th>Co-author Network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$A \rightarrow B$</td>
<td>$B \rightarrow A$</td>
</tr>
<tr>
<td></td>
<td>Macro F1</td>
<td>Accuracy</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>0.282</td>
<td>0.381</td>
</tr>
<tr>
<td>LINE</td>
<td>0.156</td>
<td>0.214</td>
</tr>
<tr>
<td>Node2vec</td>
<td>0.147</td>
<td>0.196</td>
</tr>
<tr>
<td>GraphSAGE Unsup</td>
<td>0.671</td>
<td>0.703</td>
</tr>
<tr>
<td>DANE</td>
<td><strong>0.797</strong></td>
<td><strong>0.803</strong></td>
</tr>
</tbody>
</table>
Comparison with the Variant without adversarial learning
Main Contents

- Background
- Graph embedding with hierarchical community structure
- Domain adaptive graph embedding
- Future works
Future Works

- Understanding of graph neural networks
  - Why does it work?
  - What kind of graph is it more effective?
- Customized GNN for different kinds of graphs
- Applications
  - Semi-structured data mining
  - Source code analytics
Welcome to collaboration or internship!

lun.du@microsoft.com