Semi-supervised Information Network Embedding

Meng Qu, Rui Jiao, Yizhi Zhu
Information Network Analysis

- Information networks are ubiquitous
  - Social network, citation network, PPI network
- Information network analysis
  - Node classification, link prediction, visualization

- Input of most information network analysis models
  - Low-dimensional feature vectors of nodes

- How to generate the feature vectors?
Information Network Embedding

• Goals of information network embedding models
  - Learn low-dimensional vector representations of each vertex

• Basic idea
  - Preserve structure information of the network
  - Exploit 1\textsuperscript{st} and 2\textsuperscript{nd} order proximities between vertices
  - Similar vertices will have similar embedding vectors

6 & 7 are similar: high 1\textsuperscript{st} order similarity
5 & 6 are also similar: high 2\textsuperscript{nd} order similarity
Limitation

• Classical embedding models are completely unsupervised
  - General for various tasks
  - Weak for particular predictive task

• In most tasks we may have additional labels
  - Wikipedia pages: category labels
  - DBLP papers: conference labels

• Labeling information
  - Complementary to structure information
  - Provide guidance for embedding learning
Existing Semi-supervised Embedding Models

• Formulate the problem as a matrix factorization problem

• The objective functions consist of two parts:
  - Maximize the probability of the network structure
  - Minimize the predictive error on labeled vertices

\[ \sum_{(u,v) \in E} L(u,v) + \sum_{v \in V_L} L(v, y_v) \]

• Weak points:
  - Bad efficiency:
    - the adjacency matrix of a network could be very large
  - Bad effectiveness:
    - the labeled data could be very sparse
    - The labeling signal is too weak
Motivation

• Nodes turn to have the same labels as their neighbors
  - Social networks: a user and his/her followers turn to have similar interests
  - Co-author networks: authors of a paper may have similar research interests

• Our solution
  - Propagate the labeling information to the whole network based on random walk
  - Learn vertex embeddings based on the dense labeling information
SemiEmbed: Unsupervised Part

- The objective function:
  - Unsupervised part: maximize the probability of the network structure
  - Supervised part: minimize the predictive error

- Unsupervised part:
  - Extend LINE to consider high-order proximities
  - In each iteration, sample two edges (a,b) and (b,c)
  - Treat both b and c as the neighbor of a
SemiEmbed: Supervised Part

- Supervised part:
  - Randomly sample a labeled vertex $v$ and suppose its label is $l$
  - Do random walk from $v$ with termination probability as $p$
  - Suppose the walk path is $(v, v_1, v_2, ..., v_m)$, then we assume all $v_i$ are associated with $l$ and we maximize $\log \sigma(v_i, l)$.
Framework of SemiEmbed

Data: \( G = (V, E) \), a set of labeled vertices \( V_L \), the termination probability as \( p \), number of training samples \( S \), number of negative samples \( N \), and the initial learning rate \( \alpha \).

Result: vertex embeddings \( \mathbf{x}_v \).

while \( \text{iter} \leq S \) do

/* Use Structure Information */
Randomly sample an edge \((u, v) \in E\) with the probability proportional to the edge weights.
Randomly sample an edge \((v, w) \in E\) with the probability proportional to the edge weights.
Update \( \mathbf{x}_u, \mathbf{c}_v \) to maximize \( \log \sigma(\mathbf{x}_v^T \mathbf{c}_v) \).
Update \( \mathbf{x}_v, \mathbf{c}_w \) to maximize \( \log \sigma(\mathbf{x}_w^T \mathbf{c}_w) \).

while \( k \leq N \) do

Randomly sample a vertex \( v_n \) with the probability proportional to vertex degree.
Update \( \mathbf{x}_u, \mathbf{c}_{v_n} \) to maximize \( \log \sigma(-\mathbf{x}_u^T \mathbf{c}_{v_n}) \).
end

/* Use Labeling Information */
Randomly sample a labeled vertex \( v_l \) and denote its label as \( l \).
Do random walk from \( v_l \) with termination probability as \( p \) and denote the obtained path as \((v, v_1, v_2, \ldots, v_m)\).
while \( l \leq m \) do

Update \( \mathbf{x}_l, \mathbf{x}_{v_n} \) to maximize \( \log \sigma(\mathbf{x}_l^T \mathbf{x}_{v_n}) \).
while \( k \leq N \) do

Randomly sample a vertex \( v_n \) with the probability proportional to vertex degree.
Update \( \mathbf{x}_l, \mathbf{c}_{v_n} \) to maximize \( \log \sigma(-\mathbf{x}_l^T \mathbf{c}_{v_n}) \).
end
end

Unsupervised Part

Supervised Part
Experiments

• Datasets:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Citeseer</th>
<th>Wiki</th>
<th>Youtube</th>
<th>PubMed</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Vertices</td>
<td>3,312</td>
<td>2,405</td>
<td>1,138,499</td>
<td>566,760</td>
</tr>
<tr>
<td>#Edges</td>
<td>4,732</td>
<td>17,981</td>
<td>2,990,443</td>
<td>265,286,440</td>
</tr>
<tr>
<td>#Labels</td>
<td>6</td>
<td>17</td>
<td>47</td>
<td>116</td>
</tr>
<tr>
<td>#Labeled Data</td>
<td>3,312</td>
<td>2,405</td>
<td>1,138,499</td>
<td>56,341</td>
</tr>
</tbody>
</table>

• Compared algorithms:
  - LINE (unsupervised), DeepWalk (unsupervised)
  - LINE-Ext: unsupervised part of SemiEmbed (unsupervised)
  - MMDW (semi-supervised)
  - SemiEmbed (no walk): SemiEmbed with termination probability as 1
  - SemiEmbed (semi-supervised)
Results on Small Dataset

Table 2: Results of vertex classification on CITESEER network.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Algorithm</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
<th>60%</th>
<th>70%</th>
<th>80%</th>
<th>90%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>DeepWalk</td>
<td>49.09</td>
<td>55.96</td>
<td>60.65</td>
<td>63.97</td>
<td>65.42</td>
<td>67.29</td>
<td>66.80</td>
<td>66.82</td>
<td>63.91</td>
</tr>
<tr>
<td></td>
<td>LINE</td>
<td>39.82</td>
<td>46.83</td>
<td>49.02</td>
<td>50.65</td>
<td>53.77</td>
<td>54.20</td>
<td>53.87</td>
<td>54.67</td>
<td>53.82</td>
</tr>
<tr>
<td></td>
<td>LINE-Ext</td>
<td>44.97</td>
<td>51.96</td>
<td>54.21</td>
<td>55.62</td>
<td>59.03</td>
<td>59.31</td>
<td>59.34</td>
<td>59.28</td>
<td>59.37</td>
</tr>
<tr>
<td></td>
<td>MMDW</td>
<td>55.60</td>
<td>60.97</td>
<td>63.18</td>
<td>65.08</td>
<td>66.93</td>
<td>69.52</td>
<td>70.47</td>
<td>70.87</td>
<td>70.95</td>
</tr>
<tr>
<td></td>
<td>SemiEmb(no walk)</td>
<td>45.01</td>
<td>51.90</td>
<td>54.13</td>
<td>55.70</td>
<td>59.05</td>
<td>59.21</td>
<td>59.31</td>
<td>59.23</td>
<td>59.29</td>
</tr>
<tr>
<td></td>
<td>SemiEmb</td>
<td><strong>57.71</strong></td>
<td><strong>63.10</strong></td>
<td><strong>65.37</strong></td>
<td><strong>67.35</strong></td>
<td><strong>68.06</strong></td>
<td><strong>71.79</strong></td>
<td><strong>73.24</strong></td>
<td><strong>64.75</strong></td>
<td><strong>74.02</strong></td>
</tr>
</tbody>
</table>

• LINE-Ext, DeepWalk > LINE
  - The network is very sparse

• LINE-Ext = SemiEmbed (no walk)
  - Only minimizing the predictive error on labeled vertices lead to marginal improvements

• SemiEmbed > SemiEmbed (no walk), MMDW
  - Our propagation strategy is very effective
Results on Small Dataset

Table 3: Results of vertex classification on WIKI network.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Algorithm</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
<th>60%</th>
<th>70%</th>
<th>80%</th>
<th>90%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>DeepWalk</td>
<td>52.03</td>
<td>54.62</td>
<td>59.80</td>
<td>60.29</td>
<td>61.26</td>
<td>65.41</td>
<td>65.84</td>
<td>66.53</td>
<td>68.16</td>
</tr>
<tr>
<td></td>
<td>LINE</td>
<td>52.17</td>
<td>53.62</td>
<td>57.81</td>
<td>57.26</td>
<td>58.94</td>
<td>62.46</td>
<td>62.24</td>
<td>66.74</td>
<td>67.35</td>
</tr>
<tr>
<td></td>
<td>LINE-Ext</td>
<td>53.02</td>
<td>53.70</td>
<td>57.97</td>
<td>57.36</td>
<td>59.01</td>
<td>63.01</td>
<td>62.81</td>
<td>67.04</td>
<td>67.64</td>
</tr>
<tr>
<td></td>
<td>MMDW</td>
<td>57.76</td>
<td>62.34</td>
<td>65.76</td>
<td>67.31</td>
<td>67.33</td>
<td>68.97</td>
<td>70.12</td>
<td>72.82</td>
<td>74.29</td>
</tr>
<tr>
<td></td>
<td>SemiEmb(no walk)</td>
<td>54.85</td>
<td>55.20</td>
<td>58.13</td>
<td>57.91</td>
<td>59.79</td>
<td>63.22</td>
<td>62.79</td>
<td>67.03</td>
<td>67.92</td>
</tr>
<tr>
<td></td>
<td>SemiEmb</td>
<td><strong>59.52</strong></td>
<td><strong>63.83</strong></td>
<td><strong>66.84</strong></td>
<td><strong>68.32</strong></td>
<td><strong>68.23</strong></td>
<td><strong>69.21</strong></td>
<td><strong>70.17</strong></td>
<td>72.71</td>
<td>73.86</td>
</tr>
</tbody>
</table>

- **LINE = LINE-Ext = DeepWalk**
  - Dense network

- **SemiEmbed > MMDW**
  - When the number of labeling data is small
  - The propagation strategy is more effective when the labeling information is sparse
Results on Large Datasets

- SemiEmbed is very efficient and effective on large datasets

Table 4: Results of vertex classification on large network. As the networks are too large for some methods, some results are missing here.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Algorithm</th>
<th>Youtube</th>
<th>PubMed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>DeepWalk</td>
<td>45.23</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>LINE</td>
<td>43.34</td>
<td>51.79</td>
</tr>
<tr>
<td></td>
<td>LINE-Ext</td>
<td>45.82</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MMDW</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>SemiEmb(no walk)</td>
<td>45.87</td>
<td>51.67</td>
</tr>
<tr>
<td></td>
<td>SemiEmb</td>
<td>47.39</td>
<td>54.42</td>
</tr>
</tbody>
</table>
Performances w.r.t. Termination Probability

• Performance increases as we decrease the termination probability

(a) Citeseer

(b) Wiki
Conclusion

• Limitation of existing work
  - Unsupervised methods: lack of guidance from labeled data
  - Semi-supervised methods: not efficient, may not effectively use labeled data

• Our proposed SemiEmbed model
  - Propagate labeling information to the whole network
  - Learn embedding guided by the dense labeling information and structure information
  - Jointly execute the propagation process and the learning process
  - Experimental results demonstrate its effectiveness and efficiency
Thanks!