Representation Learning on Networks

Yuxiao Dong
Microsoft Research, Redmond

Joint work with Jiezhong Qiu, Jie Zhang, Jie Tang (Tsinghua University) Hao Ma (MSR & Facebook AI) and Kuansan Wang (MSR)
Networks

Social networks

Economic networks

Biomedical networks

Information networks

Internet

Networks of neurons

Slides credit: Jure Leskovec
The Network & Graph Mining Paradigm

Graph & network applications
- Node label inference;
- Link prediction;
- User behavior… …

$x_{ij}$: node $v_i$’s $j^{th}$ feature, e.g., $v_i$’s pagerank value

hand-crafted feature matrix

Handcrafted feature matrix

feature engineering

machine learning models
Feature engineering learning

- Input: a network $G = (V, E)$
- Output: $Z \in R^{|V| \times k}$, $k \ll |V|$, $k$-dim vector $Z_v$ for each node $v$. 

Graph & network applications
- Node label inference;
- Node clustering;
- Link prediction;
  - … …
Network Embedding: Random Walk + Skip-Gram

- sentences in NLP
- vertex-paths in Networks

skip-gram (word2vec)

Random Walk Strategies

- Random Walk
  - DeepWalk (walk length > 1)
  - LINE (walk length = 1)

- Biased Random Walk
  - 2\textsuperscript{nd} order Random Walk
    - node2vec
  - Metapath guided Random Walk
    - metapath2vec
Application: Embedding Heterogeneous Academic Graph

- Microsoft Academic Graph
  - 225,572,477 Papers
  - 244,499,947 Authors
  - 664,891 Topics
  - 4,397 Conferences
  - 48,758 Journals
  - 25,554 Institutions


- [https://academic.microsoft.com/](https://academic.microsoft.com/)
- [https://www.openacademic.ai/oag/](https://www.openacademic.ai/oag/)
Application 1: Related Venues

- [https://academic.microsoft.com/](https://academic.microsoft.microsoft.com/)
- [https://www.openacademic.ai/oag/](https://www.openacademic.ai/oag/)
Application 2: Similarity Search (Institution)

- https://academic.microsoft.com/
- https://www.openacademic.ai/oag/
Network Embedding

Input:
Adjacency Matrix $A$

Output:
Vectors $Z$

- Random Walk
  - DeepWalk (walk length > 1)
  - LINE (walk length = 1)
- Biased Random Walk
  - 2nd order Random Walk
    - node2vec
  - Metapath guided Random Walk
    - metapath2vec
Unifying DeepWalk, LINE, PTE, & node2vec as Matrix Factorization

- **DeepWalk**
  \[
  \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right)
  \]

- **LINE**
  \[
  \log \left( \frac{\text{vol}(G)}{b} D^{-1} A D^{-1} \right)
  \]

- **PTE**
  \[
  \log \left( \begin{bmatrix}
  \alpha \text{vol}(G_{ww})(D_{row}^{ww})^{-1} A_{ww}(D_{col}^{ww})^{-1} \\
  \beta \text{vol}(G_{dw})(D_{row}^{dw})^{-1} A_{dw}(D_{col}^{dw})^{-1} \\
  \gamma \text{vol}(G_{lw})(D_{row}^{lw})^{-1} A_{lw}(D_{col}^{lw})^{-1}
  \end{bmatrix} \right) - \log b
  \]

- **node2vec**
  \[
  \log \left( \frac{1}{2T} \sum_{r=1}^{T} \left( \frac{\sum_u X_{w,u} P_{c,w,u}^r + \sum_u X_{c,u} P_{w,c,u}^r}{b (\sum_u X_{w,u}) (\sum_u X_{c,u})} \right) \right)
  \]

\[A \text{ Adjacency matrix} \quad b: \#\text{negative samples} \quad T: \text{context window size}\]

\[\text{vol}(G) = \sum_i \sum_j A_{ij}\]
$G = (V,E)$
- Adjacency matrix $A$
- Degree matrix $D$
- Volume of $G$: $vol(G)$

\[
\log\left( \frac{\#(w, c)|D|}{b\#(w)#(c)} \right)
\]

- $(w, c)$: co-occurrence of $w$ & $c$
- $(w)$: occurrence of node $w$
- $(c)$: occurrence of context $c$
- $D$: node-context pair $(w, c)$ multi-set
- $|D|$: number of node-context pairs

Levy and Goldberg. Neural word embeddings as implicit matrix factorization. In *NIPS 2014*
Understanding Random Walk + Skip Gram

Suppose the multiset \((\mathcal{W}, \mathcal{C}) \mid \mathcal{D}\) is constructed based on random walk on \(\mathcal{D}\).

- \((w, c)\): co-occurrence of node \(w\) and context \(c\)
- \((w)\): occurrence of node \(w\)
- \((c)\): occurrence of context \(c\)
- \(\mathcal{D}\): node–context pair \((w, c)\) multi-set
- \(|\mathcal{D}|\): number of node-context pairs

\[
\log \left( \frac{\#(w, c) |\mathcal{D}|}{b \#(w) \#(c)} \right)
\]
Understanding Random Walk + Skip Gram

- Partition the multiset $\mathcal{D}$ into several sub-multisets according to the way in which each node and its context appear in a random walk node sequence.
- More formally, for $r = 1, 2, \cdots, T$, we define

  $\mathcal{D}_r = \{(w, c) : (w, c) \in \mathcal{D}, w = w_{j+r}^n, c = w_{j+r}^n\}$

  $\mathcal{D}_{\overline{r}} = \{(w, c) : (w, c) \in \mathcal{D}, w = w_{j+r}^n, c = w_j^n\}$

Distinguish direction and distance

- $(w, c)$: co-occurrence of $w$ & $c$
- $(w)$: occurrence of node $w$
- $(c)$: occurrence of context $c$
- $\mathcal{D}$: node-context pair $(w, c)$ multi-set
- $|\mathcal{D}|$: number of node-context pairs

\[
\log\left(\frac{\#(w, c)|\mathcal{D}|}{b\#(w)\#(c)}\right)
\]
Understanding Random Walk + Skip Gram

\[ \log \left( \frac{\#(w, c) |\mathcal{D}|}{b \#(w) \cdot \#(c)} \right) = \log \left( \frac{b \#(w, c)}{\#(w) \cdot \#(c)} \right) \]

the length of random walk \( L \to \infty \)

- \((w, c): \) co-occurrence of \( w \) & \( c \)
- \( \mathcal{D}: (w, c) \) multi-set

\[
\frac{\#(w, c)}{|\mathcal{D}|} = \frac{1}{2T} \sum_{r=1}^{T} \left( \frac{\#(w, c)_{\rightarrow}}{|\mathcal{D}_\rightarrow|} + \frac{\#(w, c)_{\leftarrow}}{|\mathcal{D}_\leftarrow|} \right)
\]

\[
\frac{\#(w, c)_{\rightarrow}}{|\mathcal{D}_\rightarrow|} \xrightarrow{p} \frac{d_w}{\text{vol}(G)} (P^r)_{w,c}
\]

\[
\frac{\#(w, c)_{\leftarrow}}{|\mathcal{D}_\leftarrow|} \xrightarrow{p} \frac{d_c}{\text{vol}(G)} (P^r)_{c,w}
\]

\[
\frac{\#(w)}{|\mathcal{D}|} \xrightarrow{p} \frac{d_w}{\text{vol}(G)}
\]

\[
\frac{\#(c)}{|\mathcal{D}|} \xrightarrow{p} \frac{d_c}{\text{vol}(G)}
\]

\[
P = D^{-1} A
\]
Understanding Random Walk + Skip Gram

\[
\log \left( \frac{\#(w, c)}{b \cdot \#(w) \cdot \#(c)} \right) = \log \left( \frac{\#(w, c)}{b \cdot \#(w) \cdot \#(c)} \right)
\]

the length of random walk \( L \to \infty \)

\[
\frac{\#(w, c)}{|\mathcal{D}|} \xrightarrow{p} \frac{1}{2T} \sum_{r=1}^{T} \left( \frac{d_w}{\text{vol}(G)} (P^r)_{w,c} + \frac{d_c}{\text{vol}(G)} (P^r)_{c,w} \right)
\]

\[
P = D^{-1} A
\]
DeepWalk is asymptotically and implicitly factorizing

\[
\log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1}A)^r \right) D^{-1} \right)
\]

**A** Adjacency matrix  
**D** Degree matrix  

\[
\text{vol}(G) = \sum_i \sum_j A_{ij}
\]

*b*: #negative samples  

*T*: context window size

1. Qiu et al. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In WSDM’18.
Unifying DeepWalk, LINE, PTE, & node2vec as Matrix Factorization

- DeepWalk
  \[ \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right) \]

- LINE
  \[ \log \left( \frac{\text{vol}(G)}{b} D^{-1} A D^{-1} \right) \]

- PTE
  \[ \log \left( \begin{bmatrix} \alpha \text{vol}(G_{ww}) (D_{ww}^{-1}) A_{ww} (D_{ww}^{-1}) \\ \beta \text{vol}(G_{dw}) (D_{dw}^{-1}) A_{dw} (D_{dw}^{-1}) \\ \gamma \text{vol}(G_{lw}) (D_{lw}^{-1}) A_{lw} (D_{lw}^{-1}) \end{bmatrix} \right) - \log b \]

- node2vec
  \[ \log \left( \frac{1}{2T} \sum_{r=1}^{T} \left( \sum_{u} X_{w,u} P_{c,w,u}^{r} + \sum_{u} X_{c,u} P_{w,c,u}^{r} \right) \right) b \left( \sum_{u} X_{w,u} \right) \left( \sum_{u} X_{c,u} \right) \]

Qiu et al. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In WSDM’18. The most cited paper in WSDM’18 as of May 2019
NetMF: explicitly factorizing the DeepWalk matrix

DeepWalk is asymptotically and implicitly factorizing

\[
\log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1}A)^r \right) D^{-1} \right)
\]

1. Qiu et al. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In *WSDM’18*. 
the NetMF algorithm

1. Construction
2. Factorization

\[ S = \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right) \]

1. Qiu et al. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In WSDM’18.
Results

- Predictive performance on varying the ratio of training data;
- The $x$-axis represents the ratio of labeled data (％)

1. Qiu et al. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In *WSDM’18*. 
Explicit matrix factorization (NetMF) offers performance gains over implicit matrix factorization (DeepWalk & LINE)

1. Qiu et al. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In WSDM’18.
Input:
Adjacency Matrix \( A \)

\[ S = f(A) \]

Output:
Vectors \( Z \)

Incorporate network structures \( A \) into the similarity matrix \( S \), and then factorize \( S \)

\[ f(A) = \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right)^{-1} \right) \]
Challenges

$\Rightarrow \quad S = \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right)$

NetMF is not practical for very large networks
NetMF

How can we solve this issue?

1. Construction
2. Factorization

\[ S = \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1}A)^r \right) D^{-1} \right) \]

How can we solve this issue?

1. **Sparse Construction**
2. **Sparse** Factorization

\[
S = \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right)
\]

Sparsify $S$

For random-walk matrix polynomial $L = D - \sum_{r=1}^{T} \alpha_r D (D^{-1} A)^r$

where $\sum_{r=1}^{T} \alpha_r = 1$ and $\alpha_r$ non-negative.

One can construct a $(1 + \epsilon)$-spectral sparsifier $\tilde{L}$ with $O(n \log n \epsilon^{-2})$ non-zeros in time $O(T^2 m \epsilon^{-2} \log^2 n)$ for undirected graphs.

Suppose $G = (V, E, A)$ and $\tilde{G} = (V, \tilde{E}, \tilde{A})$ are two weighted undirected networks. Let $L = D_G - A$ and $\tilde{L} = D_{\tilde{G}} - \tilde{A}$ be their Laplacian matrices, respectively. We define $G$ and $\tilde{G}$ are $(1 + \epsilon)$-spectrally similar if

$$\forall x \in \mathbb{R}^n, (1 - \epsilon) \cdot x^T \tilde{L} x \leq x^T L x \leq (1 + \epsilon) \cdot x^T \tilde{L} x.$$

Sparsify $S$

For random-walk matrix polynomial

$$L = D - \sum_{r=1}^{T} \alpha_r D \left(D^{-1} A\right)^r$$

where $\sum_{r=1}^{T} \alpha_r = 1$ and $\alpha_r$ non-negative

One can construct a $(1 + \epsilon)$-spectral sparsifier $\tilde{L}$ with $O(n \log n \epsilon^{-2})$ non-zeros in time $O(T^2 m \epsilon^{-2} \log^2 n)$

$$S = \log^\circ \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right)$$

$$\alpha_1 = \cdots = \alpha_T = \frac{1}{T} \quad \Rightarrow \quad \approx \log^\circ \left( \frac{\text{vol}(G)}{b} D^{-1} (D - \tilde{L}) D^{-1} \right)$$

NetSMF --- Sparse

- Construct a random walk matrix polynomial sparse matrix factorizer, $\tilde{L}$

- Construct a NetMF matrix sparsifier.

$$\text{trunc}_{\log^\circ} \left( \frac{\text{vol}(G)}{b} D^{-1} (D - \tilde{L}) D^{-1} \right)$$

- Factorize the constructed sparse matrix

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>$O(MT \log n)$ for weighted networks $O(MT)$ for unweighted networks</td>
<td>$O(M + n + m)$</td>
</tr>
<tr>
<td>Step 2</td>
<td>$O(M)$</td>
<td>$O(M + n)$</td>
</tr>
<tr>
<td>Step 3</td>
<td>$O(Md + nd^2 + d^3)$</td>
<td>$O(M + nd)$</td>
</tr>
</tbody>
</table>

NetSMF---bounded approximation error

\[
\log^\circ \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right) = \log^\circ \left( \frac{\text{vol}(G)}{b} D^{-1}(D - L)D^{-1} \right) \to M
\]

\[
\approx \log^\circ \left( \frac{\text{vol}(G)}{b} D^{-1}(D - \tilde{L})D^{-1} \right) \to \tilde{M}
\]

**Theorem**

The singular value of \( \tilde{M} - M \) satisfies

\[
\sigma_i(\tilde{M} - M) \leq \frac{4\epsilon}{\sqrt{d_i d_{\min}}}, \forall i \in [n].
\]

**Theorem**

Let \( \| \cdot \|_F \) be the matrix Frobenius norm. Then

\[
\| \text{trunc}_{\cdot} \log^\circ \left( \frac{\text{vol}(G)}{b} \tilde{M} \right) - \text{trunc}_{\cdot} \log^\circ \left( \frac{\text{vol}(G)}{b} M \right) \|_F \leq \frac{4\epsilon \text{vol}(G)}{b \sqrt{d_{\min}}} \sqrt{\sum_{i=1}^{n} \frac{1}{d_i}}.
\]

#non-zeros

\(~4.5\text{ Quadrillion} \rightarrow 45\text{ Billion}\)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>BlogCatalog</th>
<th>PPI</th>
<th>Flickr</th>
<th>YouTube</th>
<th>OAG</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>V</td>
<td>$</td>
<td>10,312</td>
<td>3,890</td>
<td>80,513</td>
</tr>
<tr>
<td>$</td>
<td>E</td>
<td>$</td>
<td>333,983</td>
<td>76,584</td>
<td>5,899,882</td>
</tr>
<tr>
<td>#labels</td>
<td>39</td>
<td>50</td>
<td>195</td>
<td>47</td>
<td>19</td>
</tr>
</tbody>
</table>

Effectiveness:
• (sparse MF)NetSMF ≈ (explicit MF)NetMF > (implicit MF) DeepWalk/LINE

Efficiency:
• Sparse MF can handle billion-scale network embedding
Embedding Dimension?

Network Embedding

Input:
Adjacency Matrix $A$

Output:
Vectors $Z$

$S = f(A)$

Incorporate network structures $A$ into the similarity matrix $S$, and then factorize $S$

$$f(A) = \log \left( \frac{\text{vol}(G)}{b} \left( \frac{1}{T} \sum_{r=1}^{T} (D^{-1} A)^r \right) D^{-1} \right)$$
ProNE: More fast & scalable network embedding

Input: $G = (V, E)$

Fast Embedding Initialization via Sparse Matrix Factorization

Enhance Embedding via Spectral Propagation

ProNE

Output: $R_d$

Embedding enhancement via spectral propagation

\[ R_d \leftarrow D^{-1} A(I_n - \tilde{L}) R_d \]

\[ \tilde{L} = U g(\Lambda) U^T \]

is the spectral filter of \( L = I_n - D^{-1} A \)

\( D^{-1} A(I_n - \tilde{L}) \) is \( D^{-1} A \) modulated by the filter in the spectrum.

The idea of **Graph Neural Networks**

Performance

ProNE offers 10-400X speedups (1 thread vs 20 threads)

ProNE embeds 100,000,000 nodes by 1 thread: 29 hours with performance superiority

A general embedding enhancement framework

Network Embedding

Input: Adjacency Matrix $A$

$S = f(A)$

Sparsify $S$

Output: Vectors $Z$

$(dense)$ Matrix Factorization

Random Walk

Skip Gram

DeepWalk, LINE, node2vec, metapath2vec

$(sparse)$ Matrix Factorization

NetMF

NetSMF

$(sparse)$ Matrix Factorization

ProNE

Factorize $A$, and then incorporate network structures via spectral propagation
Input: Adjacency Matrix $A$

$S = f(A)$

Sparsify $S$

Random Walk

Skip Gram

DeepWalk, LINE, node2vec, metapath2vec

(dense) Matrix Factorization

NetMF: theory & better accuracy

(sparse) Matrix Factorization

NetSMF: handle billion-scale graphs

(sparse) Matrix Factorization

$Z = f(Z')$

ProNE: 10--400X speedups

Output: Vectors $Z$
References

1. Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Chi Wang, Kuansan Wang, and Jie Tang. NetSMF: Large-Scale Network Embedding as Sparse Matrix Factorization. WWW 2019.


Microsoft Academic Graph

- 664,862 fields of study
- 230 million authors
- 25,570 Institutions
- 48,757 journals
- 4,307 conferences
- 228 million papers/patents/books/preprints

https://academic.microsoft.com as of Sep. 2019
The graph data is open!
Thank you!

Papers & data & code available at https://ericdongyx.github.io/ ericdongyx@gmail.com

Joint work with Jiezhong Qiu, Jie Zhang, Jie Tang (Tsinghua University) Hao Ma (MSR & Facebook AI) and Kuansan Wang (MSR)