Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec

Jiezhong Qiu

Tsinghua University

February 21, 2018

Joint work with Yuxiao Dong (MSR), Hao Ma (MSR),
Jian Li (IIIS, Tsinghua), Kuansan Wang (MSR),
Jie Tang (DCST, Tsinghua)

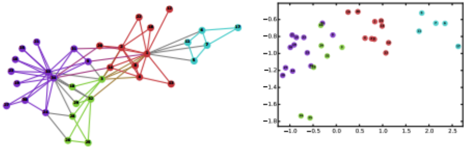
Motivation and Problem Formulation

Problem Formulation

Give a network $G = (V, E)$, aim to learn a function $f : V \to \mathbb{R}^p$ to capture neighborhood similarity and community membership.

Applications:

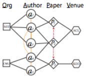
- link prediction
- community detection
- label classification



(a) Input: Karate Graph (b) Output: Representation

Figure 1: A toy example (Figure from DeepWalk).

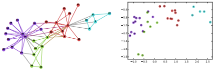
History of Network Embedding



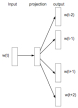
2017 metapath2vec [Dong et al.]

2016 node2vec [Grover & Leskovec]

2015 LINE & PTE [Tang et al.]



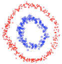
2014 DeepWalk [Perozzi et al.]



word2vec (skip-gram) [Mikolov et al.] 2013

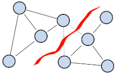
2009 SocDim [Tang & Liu]

Spectral Clustering v.s. Kernel k-means [Dhillon et al.] 2005



Spectral Clustering [Ng et al.] 2002

Image Segmentation [Shi & Malik] 2000



1996 A large body of literature
[Pothen et al.] [Simon] [Bolla],
[Hagen & Kahng] [Hendrickson & Leland]
[Van Driessche & Roose], [Barnard et al.]
[Spielman & Teng], [Guattery & Miller]

1973 Fiedler Vector [Fiedler]
Spectral Partitioning [Donath, Hoffman]

Contents

Notations

Consider an undirected weighted graph $G = (V, E)$, where $|V| = n$ and $|E| = m$.

- Adjacency matrix $\boldsymbol{A} \in \mathbb{R}_{+}^{n \times n}$:

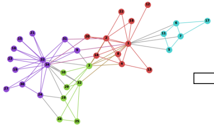
$$\boldsymbol{A}_{i,j} = \begin{cases} a_{i,j} > 0 & (i,j) \in E \\ 0 & (i,j) \notin E \end{cases}.$$

- Degree matrix $\boldsymbol{D} = \operatorname{diag}(d_1, \cdots, d_n)$, where d_i is the generalized degree of vertex i.
- Volume of the graph G: $\operatorname{vol}(G) = \sum_i \sum_j \boldsymbol{A}_{i,j}$.

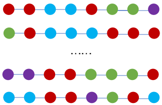
Assumption

$G = (V, E)$ is connected, undirected, and not bipartite, which makes $P(w) = \frac{d_w}{\operatorname{vol}(G)}$ a unique stationary distribution.

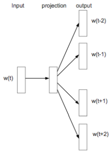
DeepWalk — Roadmap



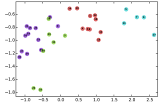
Input
G=(V,E)



Random
Walk



Skip-gram



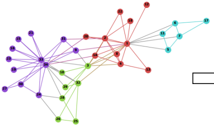
Output:
Node
Embedding

DeepWalk — a Two-step Algorithm

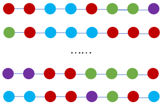
Algorithm 1: DeepWalk

for $n = 1, 2, \ldots, N$ **do**
    Pick w_1^n according to a probability distribution $P(w_1)$;
    Generate a vertex sequence $(w_1^n, \cdots, w_L^n)$ of length L by a random walk on network G;
    for $j = 1, 2, \ldots, L - T$ **do**
        for $r = 1, \ldots, T$ **do**
            Add vertex-context pair (w_j^n, w_{j+r}^n) to multiset $\mathcal{D}$;
            Add vertex-context pair (w_{j+r}^n, w_j^n) to multiset $\mathcal{D}$;
Run SGNS on $\mathcal{D}$ with b negative samples.

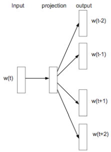
DeepWalk — Roadmap



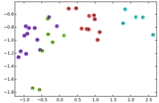
Input
G=(V,E)



Random
Walk



Skip-gram



Output:
Node
Embedding

Levy & Goldberg (NIPS 14)

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

b Number of negative samples

$\#(w,c)$ Co-occurrence of w and c

$|\mathcal{D}|$ Total number of word-context pairs

$\#(w)$ Occurrence of word w

$\#(c)$ Occurrence of context c

Skip-gram with Negative Sampling (SGNS)

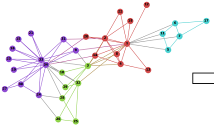
- SGNS maintains a multiset $\mathcal{D}$ which counts the occurrence of each word-context pair (w, c).
- Objective:

$$\mathcal{L} = \sum_w \sum_c \left(\#(w,c) \log g\left(\boldsymbol{x}_w^\top \boldsymbol{y}_c\right) + \frac{b\#(w)\#(c)}{|\mathcal{D}|} \log g\left(-\boldsymbol{x}_w^\top \boldsymbol{y}_c\right) \right),$$

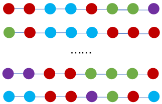
  where $\boldsymbol{x}_w, \boldsymbol{y}_c \in \mathbb{R}^d$, g is the sigmoid function, and b is the number of negative samples for SGNS.
- For sufficiently large dimensionality d, equivalent to factorizing PMI matrix (Levy & Goldberg, NIPS'14):

$$\log\left(\frac{\#(w,c)\,|\mathcal{D}|}{b\#(w)\#(c)}\right).$$

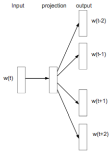
DeepWalk — Roadmap



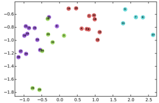
Input
G=(V,E)



Random
Walk



Skip-gram



Output:
Node
Embedding

Levy & Goldberg (NIPS 14)

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

b Number of negative samples

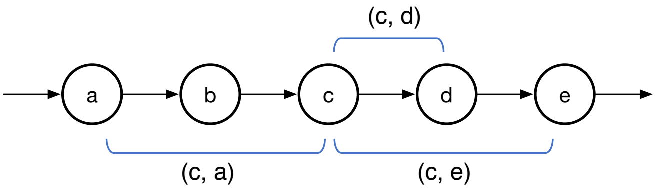
$\#(w,c)$ Co-occurrence of w and c

$|\mathcal{D}|$ Total number of word-context pairs

$\#(w)$ Occurrence of word w

$\#(c)$ Occurrence of context c

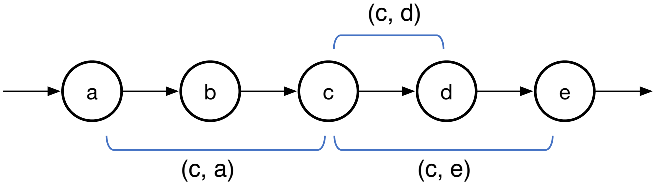
DeepWalk



Question

Suppose the multiset $\mathcal{D}$ is constructed based on random walk on graph, can we interpret $\log\left(\frac{\#(w,c)|\mathcal{D}|}{b\#(w)\#(c)}\right)$ with graph theory terminologies?

DeepWalk



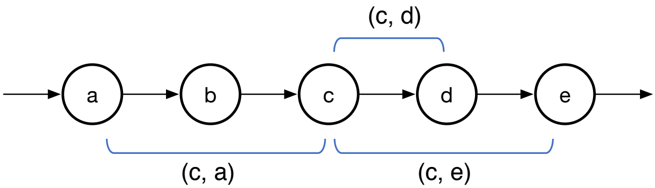
Question

Suppose the multiset $\mathcal{D}$ is constructed based on random walk on graph, can we interpret $\log\left(\frac{\#(w,c)|\mathcal{D}|}{b\#(w)\#(c)}\right)$ with graph theory terminologies?

Challange

We mix so many things together, i.e., direction and distance.

DeepWalk



Question

Suppose the multiset $\mathcal{D}$ is constructed based on random walk on graph, can we interpret $\log\left(\frac{\#(w,c)|\mathcal{D}|}{b\#(w)\#(c)}\right)$ with graph theory terminologies?

Challange

We mix so many things together, i.e., direction and distance.

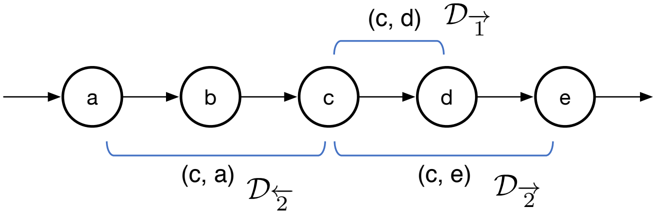
Solution

Let's distinguish them!

DeepWalk

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

$$\mathcal{D}_{\overrightarrow{r}} = \left\{(w, c) : (w, c) \in \mathcal{D}, w = w_j^n, c = w_{j+r}^n\right\},$$
$$\mathcal{D}_{\overleftarrow{r}} = \left\{(w, c) : (w, c) \in \mathcal{D}, w = w_{j+r}^n, c = w_j^n\right\}.$$



DeepWalk as Implicit Matrix Factorization

Some observations

- Observation 1:

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

- Observation 2:

$$\frac{\#(w,c)}{|\mathcal{D}|}=\frac{1}{2T}\sum_{r=1}^{T}\left(\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|}+\frac{\#(w,c)_{\overleftarrow{r}}}{|\mathcal{D}_{\overleftarrow{r}}|}\right).$$

Sufficient to characterize $\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|}$ and $\frac{\#(w,c)_{\overleftarrow{r}}}{|\mathcal{D}_{\overleftarrow{r}}|}$.

DeepWalk — Theorems

Theorem

Denote $\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}$*, when the length of random walk* $L \to \infty$*,*

$$\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|} \xrightarrow{p} \frac{d_w}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{w,c} \text{ and } \frac{\#(w,c)_{\overleftarrow{r}}}{|\mathcal{D}_{\overleftarrow{r}}|} \xrightarrow{p} \frac{d_c}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{c,w}.$$

Theorem

When the length of random walk $L \to \infty$*, we have*

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

Theorem

For DeepWalk, when the length of random walk $L \to \infty$*,*

$$\frac{\#(w,c)\,|\mathcal{D}|}{\#(w)\cdot\#(c)} \xrightarrow{p} \frac{\mathrm{vol}(G)}{2T}\left(\frac{1}{d_c}\sum_{r=1}^{T}\left(\boldsymbol{P}^r\right)_{w,c} + \frac{1}{d_w}\sum_{r=1}^{T}\left(\boldsymbol{P}^r\right)_{c,w}\right).$$

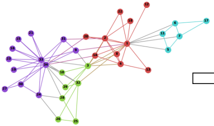
DeepWalk — Conclusion

Theorem

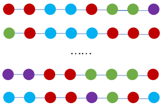
DeepWalk is asymptotically and implicitly factorizing

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right).$$

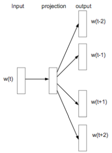
DeepWalk — Roadmap



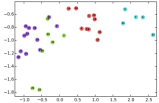
Input
G=(V,E)



Random
Walk



Skip-gram



Output:
Node
Embedding

Levy & Goldberg (NIPS 14)

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

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

$\boldsymbol{A}$ Adjacency matrix

$\boldsymbol{D}$ Degree matrix

$\mathrm{vol}(G)=\sum_i\sum_j \boldsymbol{A}_{i,j}$

b Number of negative samples

LINE

- Objective of LINE:

$$\mathcal{L} = \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \left(\boldsymbol{A}_{i,j} \log g\left(\boldsymbol{x}_i^\top \boldsymbol{y}_j\right) + \frac{b d_i d_j}{\operatorname{vol}(G)} \log g\left(-\boldsymbol{x}_i^\top \boldsymbol{y}_j\right) \right).$$

- Align it with the Objective of SGNS:

$$\mathcal{L} = \sum_{w} \sum_{c} \left(\#(w,c) \log g\left(\boldsymbol{x}_w^\top \boldsymbol{y}_c\right) + \frac{b\#(w)\#(c)}{|\mathcal{D}|} \log g\left(-\boldsymbol{x}_w^\top \boldsymbol{y}_c\right) \right).$$

- LINE is actually factorizing

$$\log\left(\frac{\operatorname{vol}(G)}{b} \boldsymbol{D}^{-1} \boldsymbol{A} \boldsymbol{D}^{-1}\right)$$

- Recall DeepWalk's matrix form:

$$\log\left(\frac{\operatorname{vol}(G)}{b} \left(\frac{1}{T} \sum_{r=1}^{T} \left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^r\right) \boldsymbol{D}^{-1}\right).$$

Observation LINE is a special case of DeepWalk ($T = 1$).

PTE

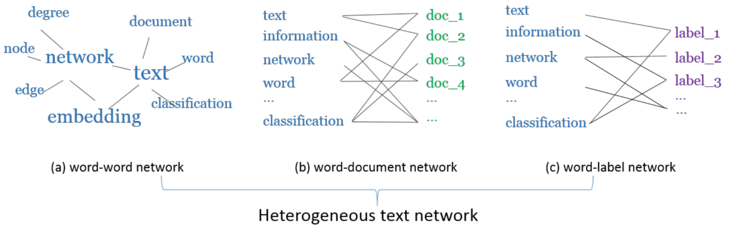


Figure 2: Heterogeneous Text Network.

- word-word network G_{ww}, $\boldsymbol{A}_{\text{ww}} \in \mathbb{R}^{\#\text{word}\times\#\text{word}}$.
- document-word network G_{dw}, $\boldsymbol{A}_{\text{dw}} \in \mathbb{R}^{\#\text{doc}\times\#\text{word}}$.
- label-word network G_{lw}, $\boldsymbol{A}_{\text{lw}} \in \mathbb{R}^{\#\text{label}\times\#\text{word}}$.

PTE as Implicit Matrix Factorization

$$\log \left(\begin{bmatrix} \alpha \operatorname{vol}(G_{\text{ww}}) (\boldsymbol{D}_{\text{row}}^{\text{ww}})^{-1} \boldsymbol{A}_{\text{ww}} (\boldsymbol{D}_{\text{col}}^{\text{ww}})^{-1} \\ \beta \operatorname{vol}(G_{\text{dw}}) (\boldsymbol{D}_{\text{row}}^{\text{dw}})^{-1} \boldsymbol{A}_{\text{dw}} (\boldsymbol{D}_{\text{col}}^{\text{dw}})^{-1} \\ \gamma \operatorname{vol}(G_{\text{lw}}) (\boldsymbol{D}_{\text{row}}^{\text{lw}})^{-1} \boldsymbol{A}_{\text{lw}} (\boldsymbol{D}_{\text{col}}^{\text{lw}})^{-1} \end{bmatrix} \right) - \log b,$$

- The matrix is of shape $(\#\text{word} + \#\text{doc} + \#\text{label}) \times \#\text{word}$.
- b is the number of negative samples in training.
- $\{\alpha, \beta, \gamma\}$ are hyper-parameters to balance the weights of the three networks. In PTE, $\{\alpha, \beta, \gamma\}$ satisfy

$$\alpha \operatorname{vol}(G_{\text{ww}}) = \beta \operatorname{vol}(G_{\text{dw}}) = \gamma \operatorname{vol}(G_{\text{lw}})$$

node2vec — 2nd Order Random Walk

$$\underline{\boldsymbol{T}}_{u,v,w} = \begin{cases} \frac{1}{p} & (u,v) \in E, (v,w) \in E, u = w; \\ 1 & (u,v) \in E, (v,w) \in E, u \neq w, (w,u) \in E; \\ \frac{1}{q} & (u,v) \in E, (v,w) \in E, u \neq w, (w,u) \notin E; \\ 0 & \text{otherwise.} \end{cases}$$

$$\underline{\boldsymbol{P}}_{u,v,w} = \operatorname{Prob}\left(w_{j+1} = u | w_j = v, w_{j-1} = w\right) = \frac{\underline{\boldsymbol{T}}_{u,v,w}}{\sum_u \underline{\boldsymbol{T}}_{u,v,w}}.$$

Stationary Distribution

$$\sum_w \underline{\boldsymbol{P}}_{u,v,w} \boldsymbol{X}_{v,w} = \boldsymbol{X}_{u,v}$$

Existence guaranteed by Perron-Frobenius theorem, but may not be unique.

node2vec as Implicit Matrix Factorization

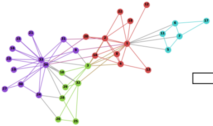
Theorem

node2vec is asymptotically and implicitly factorizing a matrix whose entry at w-th row, c-th column is

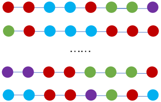
$$\log\left(\frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\sum_u \boldsymbol{X}_{w,u}\underline{\boldsymbol{P}}^r_{c,w,u}+\sum_u \boldsymbol{X}_{c,u}\underline{\boldsymbol{P}}^r_{w,c,u}\right)}{b\left(\sum_u \boldsymbol{X}_{w,u}\right)\left(\sum_u \boldsymbol{X}_{c,u}\right)}\right)$$

Contents

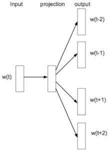
Roadmap



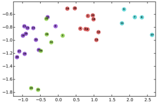
Input
G=(V,E)



Random
Walk



Skip-gram



Output:
Node
Embedding

Levy & Goldberg (NIPS 14)

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

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

Matrix
Factorization

NetMF

- Factorize the DeepWalk matrix:

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right).$$

- For numerical reason, we use truncated logarithm — $\tilde{\log}(x) = \log\left(\max(1, x)\right)$

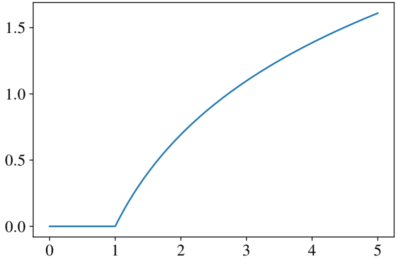


Figure 3: Truncated Logarithm

NetMF for a Small Window Size T

Algorithm 2: NetMF for a Small Window Size T

Compute $\boldsymbol{P}^1, \cdots, \boldsymbol{P}^T$;
Compute $\boldsymbol{M} = \frac{\mathrm{vol}(G)}{bT}\left(\sum_{r=1}^{T} \boldsymbol{P}^r\right)\boldsymbol{D}^{-1}$;
Compute $\boldsymbol{M}' = \max(\boldsymbol{M}, 1)$;
Rank-d approximation by SVD: $\log \boldsymbol{M}' = \boldsymbol{U}_d \boldsymbol{\Sigma}_d \boldsymbol{V}_d^\top$;
return $\boldsymbol{U}_d \sqrt{\boldsymbol{\Sigma}_d}$ as network embedding.

NetMF for a Large Window Size T — Observations

- We want to factorize

$$\tilde{\log}\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right).$$

- We know the property of normalized graph Laplacian

$$\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}=\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\top}$$

  where $\Lambda=\mathrm{diag}(\lambda_1,\cdots,\lambda_n)$ and $\forall\lambda_i\in[-1,1]$.

$$\begin{aligned}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}&=\left(\boldsymbol{D}^{-1/2}\right)\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\right)^{r}\right)\left(\boldsymbol{D}^{-1/2}\right)\\&=\left(\boldsymbol{D}^{-1/2}\right)\left(\boldsymbol{U}\underbrace{\left(\frac{1}{T}\sum_{r=1}^{T}\boldsymbol{\Lambda}^{r}\right)}_{\text{A polynomial}}\boldsymbol{U}^{\top}\right)\left(\boldsymbol{D}^{-1/2}\right)\end{aligned}$$

NetMF for a Large Window Size T — Observations

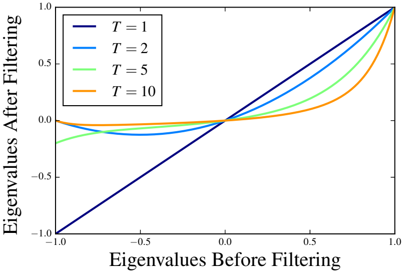


Figure 4: $f(\lambda) = \frac{1}{T} \sum_{r=1}^{T} \lambda^r$

Idea

This polynomial implicitly filters out negative eigenvalues and small positive eigenvalues, why not do it explicitly.

NetMF for a Large Window Size T — Algorithm

Algorithm 3: NetMF for a Large Window Size T

Eigen-decomposition $\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2} \approx \boldsymbol{U}_h\boldsymbol{\Lambda}_h\boldsymbol{U}_h^\top$;

Approximate $\boldsymbol{M}$ with
$\hat{\boldsymbol{M}} = \frac{\mathrm{vol}(G)}{b}\boldsymbol{D}^{-1/2}\boldsymbol{U}_h\left(\frac{1}{T}\sum_{r=1}^{T}\boldsymbol{\Lambda}_h^r\right)\boldsymbol{U}_h^\top\boldsymbol{D}^{-1/2}$;

Compute $\hat{\boldsymbol{M}}' = \max(\hat{\boldsymbol{M}}, 1)$;

Rank-d approximation by SVD: $\log\hat{\boldsymbol{M}}' = \boldsymbol{U}_d\boldsymbol{\Sigma}_d\boldsymbol{V}_d^\top$;

return $\boldsymbol{U}_d\sqrt{\boldsymbol{\Sigma}_d}$ as network embedding.

Setup

Label Classification:

- BlogCatelog, PPI, Wikipedia, Flickr
- Logistic Regression
- NetMF $(T = 1)$ v.s. LINE
- NetMF $(T = 10)$ v.s. DeepWalk

Table 1: Statistics of Datasets.

Dataset	BlogCatalog	PPI	Wikipedia	Flickr
$\|V\|$	10,312	3,890	4,777	80,513
$\|E\|$	333,983	76,584	184,812	5,899,882
#Labels	39	50	40	195

Experimental Results

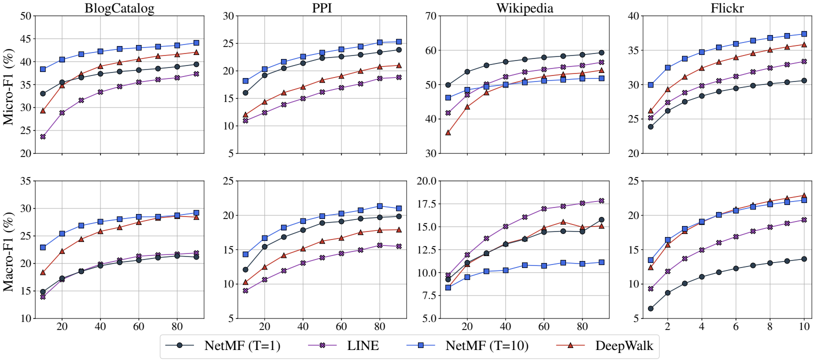


Figure 5: Predictive performance on varying the ratio of training data. The x-axis represents the ratio of labeled data (%), and the y-axis in the top and bottom rows denote the Micro-F1 and Macro-F1 scores respectively.

Conclusion

Table 2: The matrices that are implicitly approximated and factorized by DeepWalk, LINE, PTE, and node2vec.

Algorithm	Matrix
DeepWalk	$\log\left(\operatorname{vol}(G)\left(\frac{1}{T}\sum_{r=1}^{T}(\boldsymbol{D}^{-1}\boldsymbol{A})^r\right)\boldsymbol{D}^{-1}\right)-\log b$
LINE	$\log\left(\operatorname{vol}(G)\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}^{-1}\right)-\log b$
PTE	$\log\left(\begin{bmatrix}\alpha\operatorname{vol}(G_{\text{ww}})(\boldsymbol{D}_{\text{row}}^{\text{ww}})^{-1}\boldsymbol{A}_{\text{ww}}(\boldsymbol{D}_{\text{col}}^{\text{ww}})^{-1}\\ \beta\operatorname{vol}(G_{\text{dw}})(\boldsymbol{D}_{\text{row}}^{\text{dw}})^{-1}\boldsymbol{A}_{\text{dw}}(\boldsymbol{D}_{\text{col}}^{\text{dw}})^{-1}\\ \gamma\operatorname{vol}(G_{\text{lw}})(\boldsymbol{D}_{\text{row}}^{\text{lw}})^{-1}\boldsymbol{A}_{\text{lw}}(\boldsymbol{D}_{\text{col}}^{\text{lw}})^{-1}\end{bmatrix}\right)-\log b$
node2vec	$\log\left(\frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\sum_u \boldsymbol{X}_{w,u}\underline{\boldsymbol{P}}_{c,w,u}^{r}+\sum_u \boldsymbol{X}_{c,u}\underline{\boldsymbol{P}}_{w,c,u}^{r}\right)}{\left(\sum_u \boldsymbol{X}_{w,u}\right)\left(\sum_u \boldsymbol{X}_{c,u}\right)}\right)-\log b$

Thanks.

Standing on the shoulders of giants
— Isaac Newton

Code available at `github.com/xptree/NetMF`
Q&A

DeepWalk — Sketched Proof

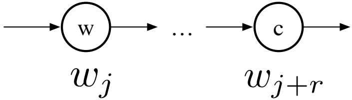
Theorem

Denote $\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}$, when $L \to \infty$, we have

$$\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|} \xrightarrow{p} \frac{d_w}{\mathrm{vol}(G)} \left(\boldsymbol{P}^r\right)_{w,c} \text{ and } \frac{\#(w,c)_{\overleftarrow{r}}}{|\mathcal{D}_{\overleftarrow{r}}|} \xrightarrow{p} \frac{d_c}{\mathrm{vol}(G)} \left(\boldsymbol{P}^r\right)_{c,w}.$$

Proof.

Consider the special case when $N = 1$, thus we only have one vertex sequence $w_1, \cdots, w_L$ generated by random walk. Let Y_j $(j = 1, \cdots, L - T)$ be the indicator function for event that $w_j = w$ and $w_{j+r} = c$



Proof (Con't)

Observation

- $\mathbb{E}[Y_j] = \text{Prob}(w_j = w, w_{j+r} = c) \to \frac{d_w}{\text{vol}(G)} \left(\boldsymbol{P}^r\right)_{w,c}$.
- $\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|} = \frac{1}{L-T} \sum_{j=1}^{L-T} Y_j$.
- $\text{Cov}(Y_i, Y_j) \to 0$ as $|i-j| \to \infty$.

Lemma

(S.N. Bernstein Law of Large Numbers) Let $Y_1, Y_2 \cdots$ be a sequence of random variables with finite expectation $\mathbb{E}[Y_j]$ and variance $\text{Var}(Y_j) < K$, $j \geq 1$, and covariances are s.t. $\text{Cov}(Y_i, Y_j) \to 0$ as $|i-j| \to \infty$. Then the law of large numbers (LLN) holds.

$$\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|} = \frac{1}{L-T} \sum_{j=1}^{L-T} Y_j \xrightarrow{p} \frac{1}{L-T} \sum_{j=1}^{L-T} \mathbb{E}(Y_j) \to \frac{d_w}{\text{vol}(G)} \left(\boldsymbol{P}^r\right)_{w,c}$$

Time Complexity

- Eigen-Decomposition (Implicitly Restarted Lanczos Method) $O(mhI + nh^2I + h^3I)$.
- Reconstruction $O(n^2h)$
- Element-wise logarithm $O(n^2)$.
- SVD (a naive implementation with eigen-decomposition): $O(n^2dI + nd^2I + d^3I)$.

Future Work

- Comprehend high-order cases, e.g., node2vec.

$$\log \left(\frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\sum_u \boldsymbol{X}_{w,u}\underline{\boldsymbol{P}}^r_{c,w,u} + \sum_u \boldsymbol{X}_{c,u}\underline{\boldsymbol{P}}^r_{w,c,u}\right)}{b\left(\sum_u \boldsymbol{X}_{w,u}\right)\left(\sum_u \boldsymbol{X}_{c,u}\right)} \right)$$

- Design scalable algorithm (e.g., using spectral sparsification of random-walk polynomials).

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^r\right)\boldsymbol{D}^{-1}\right).$$

- Connection with graph convolutional networks (Kipf & Welling, ICLR'17).