

LINE: Large-scale Information Network Embedding

2022/1/25

이종복

Content

- LINE
- Problem Definition – first, second-order proximity
- Model Description
- Model Optimization
- Experiments(results)
- Conclusion

LINE

- Embedding method in Large-scale Information Network(LINE)
 1. Designed objective function that preserves properties of the graph(local and global structures)
 2. Efficient optimization technique (should effectively find the embedding of millions of nodes – not expensive computation)

*local – first-order proximity, global – second-order proximity



Problem Definition

Problem Definition

- Information Network

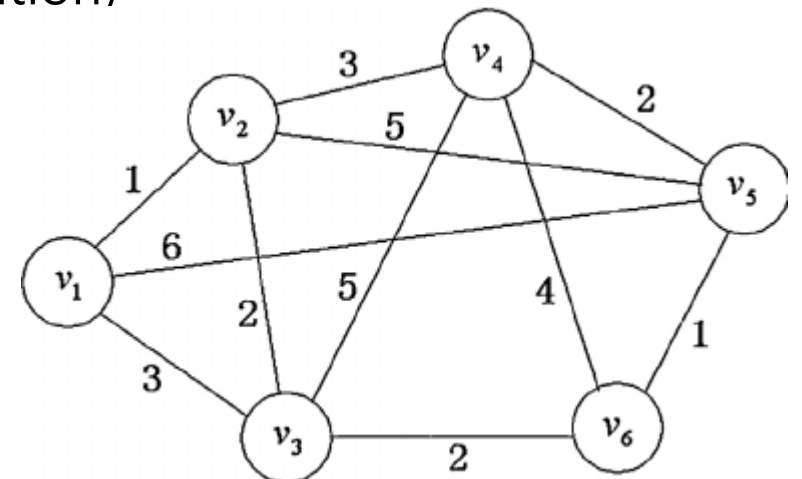
$G = (V, E)$ (set of vertices, set of edges)

$e \in E$ is an ordered pair $e = (u, v)$

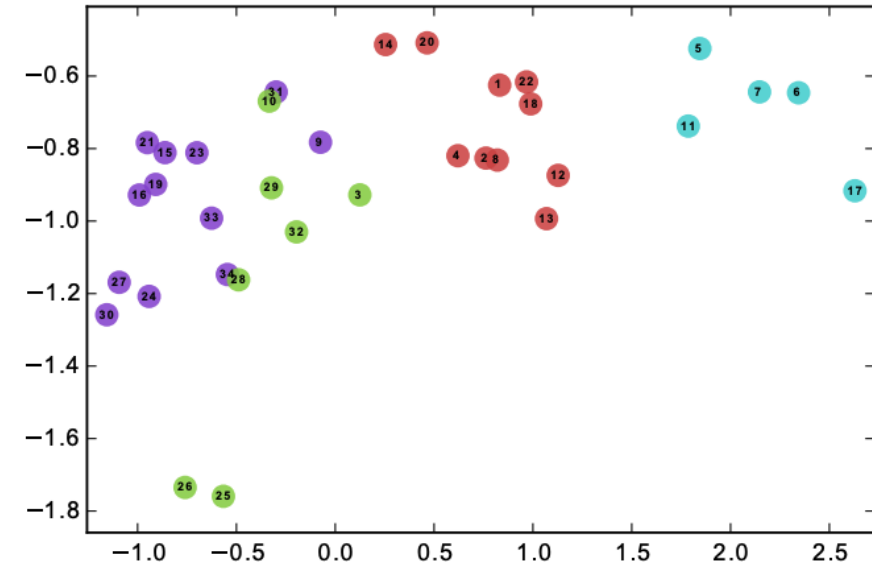
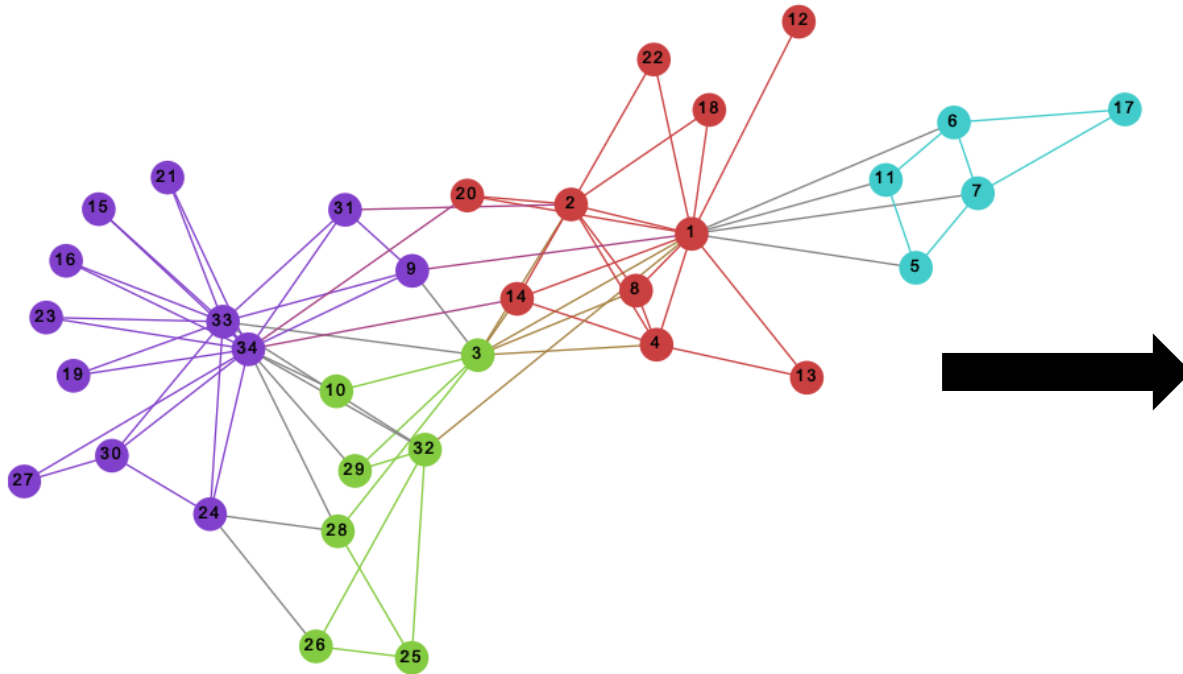
$w_{uv} > 0$ (weight which indicates the strength of relation)

Undirect: $(u, v) == (v, u)$ (has same weight)

Direct: $(u, v) != (v, u)$ (diff weight)



Problem Definition



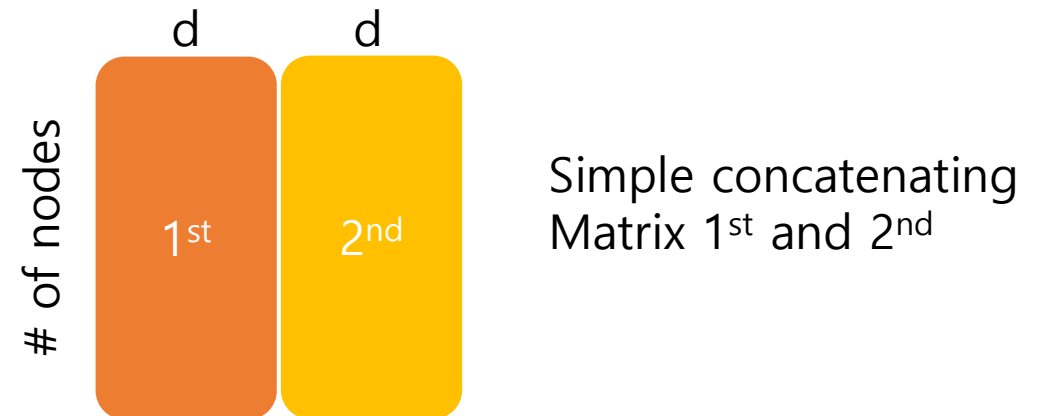
유사한 node -> 비슷한 embedding vector

기준

Local – First-order proximity

Global – Second-order proximity

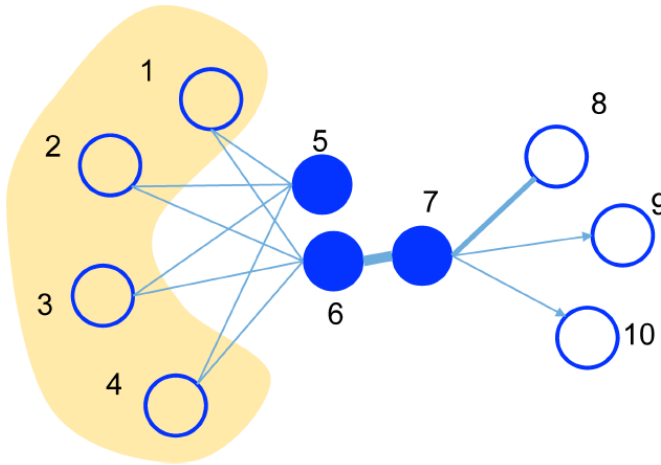
-> 두 모델을 각각 학습하여 더한다.



Problem Definition

First-order proximity

In example, vertex 6 and 7 placed closely, connected through a strong tie.
-> 6 and 7 has high first-order proximity



DEFINITION 2. (*First-order Proximity*) The **first-order** proximity in a network is the **local** pairwise proximity between two vertices. For each pair of vertices linked by an edge (u, v) , the weight on that edge, w_{uv} , indicates the first-order proximity between u and v . If no edge is observed between u and v , their **first-order** proximity is 0.

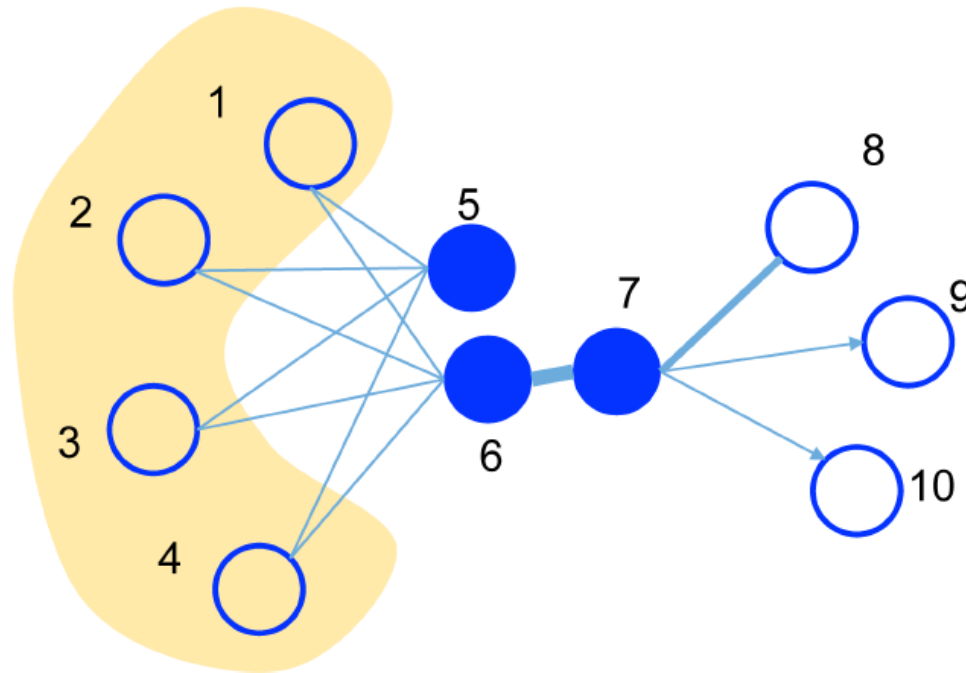
정리: W 의 크기로 표현.
 W 가 크면, 1st proximity가 큼.

Problem Definition

First-order proximity

In example, vertex 5 and 6 has zero 1st proximity.
Also in real world network, most of link missing

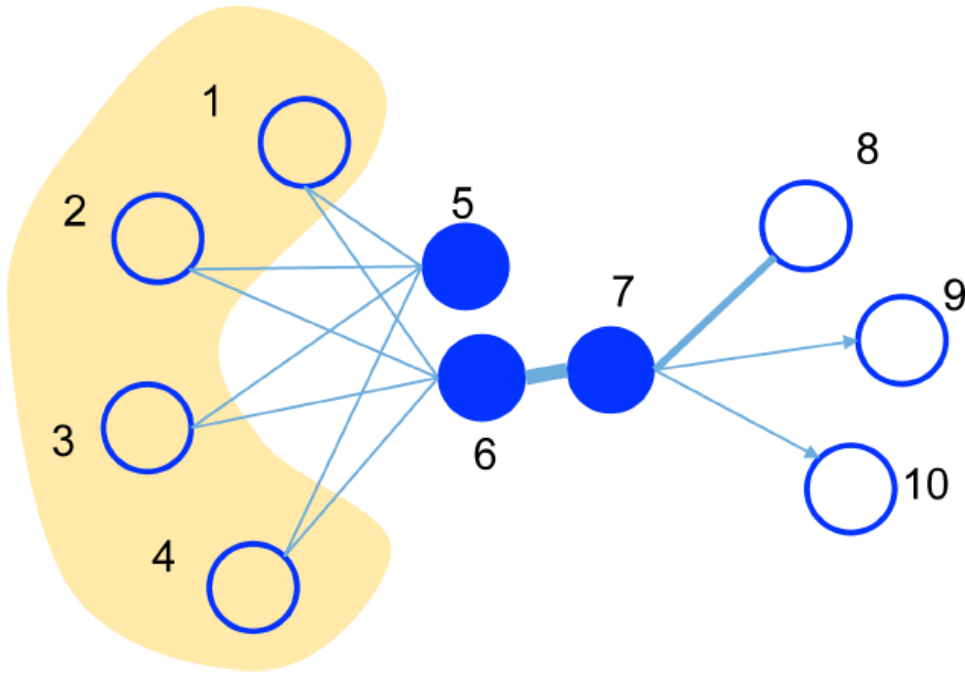
-> 1st proximity로만 object function을 사용하기에는 임베딩의 한계가 존재.



Problem Definition

Second-order proximity

In example, vertex 5 and 6 has zero 1st proximity.

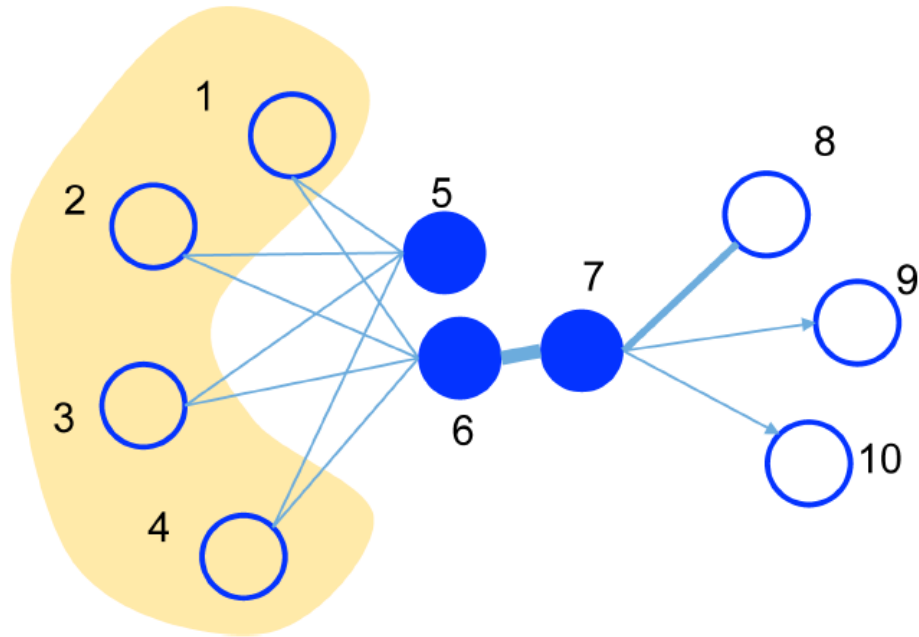


DEFINITION 3. (*Second-order Proximity*) The **second-order** proximity between a pair of vertices (u, v) in a network is the similarity between their neighborhood network structures. Mathematically, let $p_u = (w_{u,1}, \dots, w_{u,|V|})$ denote the first-order proximity of u with all the other vertices, then the second-order proximity between u and v is determined by the similarity between p_u and p_v . If no vertex is linked from/to both u and v , the **second-order** proximity between u and v is 0.

정리: 두 노드가 비슷한 neighbor를 가지고 있다면 높은 2nd proximity를 가짐.

Problem Definition

Second-order proximity



$$\hat{p}_u = (w_{u1}, w_{u2}, \dots, w_{u|V|})$$

2nd proximity: $S(\hat{p}_u, \hat{p}_v)$

*S: similarity function

예시

$$\hat{p}_5 = (1, 1, 1, 1, 0, 0, 0, 0, 0, 0)$$

$$\hat{p}_6 = (1, 1, 1, 1, 0, 0, 3, 0, 0, 0)$$

$$\hat{p}_7 = (0, 0, 0, 0, 0, 3, 0, 2, 1, 1)$$

Let S is dot product,

$$S(\hat{p}_5, \hat{p}_6) = 4$$

$$S(\hat{p}_6, \hat{p}_7) = 0$$

Model Description

Model Description

First-order proximity

Define joint probability

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-\vec{u}_i^T \cdot \vec{u}_j)}$$

Empirical probability

$$\hat{p}_1(i, j) = \frac{w_{ij}}{W}$$

Preserve 1st proximity -> minimize objective function

$$O_1 = d(\hat{p}_1(\cdot, \cdot), p_1(\cdot, \cdot))$$

We use KL-divergence as a distance function.

$$O_1 = - \sum_{(i,j) \in E} w_{ij} \log p_1(v_i, v_j)$$

***u is embedding vector

***1st proximity is only applicable for undirected graphs

Model Description

Second-order proximity

skip-gram (word embedding sota): 주변에 있는 단어들을 입력으로 중간에 있는 단어를 예측하는 방법.

중심 단어 주변 단어

↓ ↓

The fat cat sat on the mat

The fat cat sat on the mat

The fat cat sat on the mat

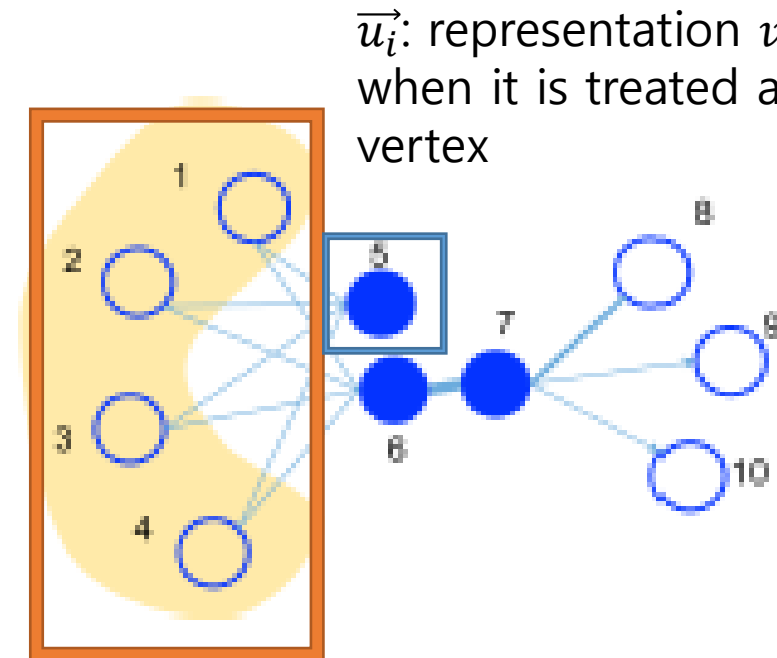
The fat cat sat on the mat

The fat cat sat on the mat

The fat cat sat on the mat

The fat cat sat on the mat

중심 단어	주변 단어
[1, 0, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 1, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 1, 0, 0]	[0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 0, 1, 0]	[0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 1]	[0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]



\vec{u}_i' : representation v_i when it is treated as a specific "context"

Model Description

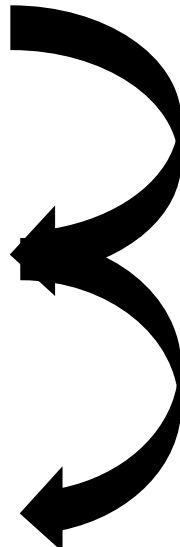
Second-order proximity

For each directed edge (i, j) , we define the probability of "context" v_j generated by vertex v_i as:

$$p_2(v_j|v_i) = \frac{\exp(\vec{u}'_j{}^T \cdot \vec{u}_i)}{\sum_{k=1}^{|V|} \exp(\vec{u}'_k{}^T \cdot \vec{u}_i)}$$

$$O_2 = \sum_{i \in V} \lambda_i d(\hat{p}_2(\cdot|v_i), p_2(\cdot|v_i))$$

$$O_2 = - \sum_{(i,j) \in E} w_{ij} \log p_2(v_j|v_i).$$



Minimize the following objective function(why?)

Adopt KL-divergence as the distance function

*** $p_2(v_j|v_i) = \frac{w_{ij}}{w_i}$ w_i means sum of node i 's neighbor.

Model Description

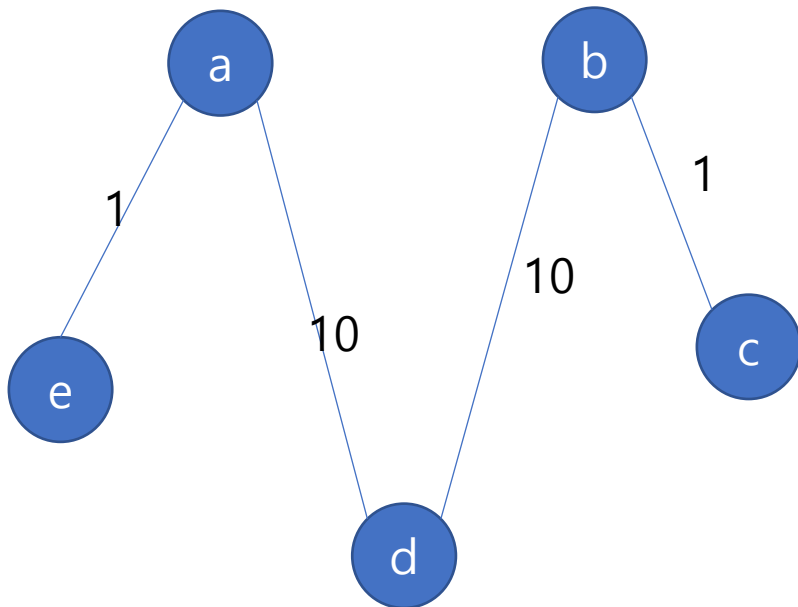
Second-order proximity

$$p_2(v_j|v_i) = \frac{\exp(\vec{u}_j'^T \cdot \vec{u}_i)}{\sum_{k=1}^{|V|} \exp(\vec{u}_k'^T \cdot \vec{u}_i)}$$

$$O_2 = \sum_{i \in V} \lambda_i d(\hat{p}_2(\cdot|v_i), p_2(\cdot|v_i))$$

$p_2(v_j|v_i) = \frac{w_{ij}}{w_i}$ w_i means sum of node i 's neighbor.

Why?



추측

a, b has high 2nd proximity.

-> a와 b의 임베딩 벡터 내적이 크도록(가깝도록) 해야함.

Objective function에 의해 a와 d가 가깝게, b와 d가 가깝게 됨.

-> a와 b가 가까워지는 결과

Model Description

정리

LINE(1st): edge weight가 큰 것을 가깝게

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-\vec{u}_i^T \cdot \vec{u}_j)}$$

$$O_1 = - \sum_{(i,j) \in E} w_{ij} \log p_1(v_i, v_j).$$

LINE(2nd): 비슷한 노드를 공유하는 노드를 가깝게

$$p_2(v_j|v_i) = \frac{\exp(\vec{u}_j^T \cdot \vec{u}_i)}{\sum_{k=1}^{|V|} \exp(\vec{u}_k^T \cdot \vec{u}_i)}$$

$$O_2 = - \sum_{(i,j) \in E} w_{ij} \log p_2(v_j|v_i).$$

Model Optimization

Model Optimization

Computation cost

LINE(1st)

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-\vec{u}_i^T \cdot \vec{u}_j)}$$



내적 1회

- > less computation
- > computationally cheap

LINE(2nd)

$$p_2(v_j|v_i) = \frac{\exp(\vec{u}_j^T \cdot \vec{u}_i)}{\sum_{k=1}^{|V|} \exp(\vec{u}_k^T \cdot \vec{u}_i)}$$



내적 $|V|$ 회

- > more computation
- > computationally expensive

Model Optimization

2nd negative sampling

Modify objective function

$$\underbrace{\log \sigma(\vec{u}_j'^T \cdot \vec{u}_i)}_{\text{Positive sample}} + \sum_{n=1}^K \underbrace{E_{v_n \sim P_n(v)} [\log \sigma(-\vec{u}_n'^T \cdot \vec{u}_i)]}_{\text{Negative sample}}$$

$P_n(v) \sim d_v^{\frac{3}{4}}$ 의 weight를 통한 sampling.(it takes $O(1)$) – Distributed representations of words and phrases and their compositionality [13]

Positive sample: observed edge (vertex에서 연결된 노드)

Negative sample: unobserved edge (연결되지 않은 노드)

-> 빠르기도 할 뿐더러, negative sample을 통해 존재하지 않는 edge에 대해서도 멀어지도록 함.

Model Optimization

Edge sampling – problem on training with ASGD(asynchronous stochastic gradient descent)

Note that the gradient will be multiplied by the weight

$$\frac{\partial O_2}{\partial \vec{u}_i} = w_{ij} \cdot \frac{\partial \log p_2(v_j | v_i)}{\partial \vec{u}_i}$$

-> it cause gradient explosion (weight의 차이에 의해 학습이 힘들.)

Ex)

weight가 극단적으로 큼 -> learning rate를 작게 함.

Weight가 극단적으로 작음 -> learning rate를 크게 함.

-> 두 해결 방법이 모순.

Model Optimization

Edge sampling – solution

Solution 1.

Edge weight에 의하여 생기는 문제

- > 모든 edge를 binary(0-1) 로 만들자 ex)weight=5인 edge를 5개의 binary edge로 연결
- > in real world network, it has large edge weight (increase memory requirement)

Solution 2.

(solution 1 적용)

일부 edge만을 뽑아(edge sampling) solution 1 을 적용한다.

이 또한 negative sampling 했던 방식과 똑같은 방식(alias table method)으로 sampling한다.

→ $O(dK|E|)$ time complexity (doesn't depend on the V)

-> improves effectiveness ASGD without compromising the efficiency

Model Optimization

Discussion

1. Low degree vertices

of neighbors가 작은 노드에 대해서는 정확한 추론이 어려움.

Intuitive solution -> adding higher order neighbors

-> consider only second-order neighbors.

$$w_{ij} = \sum_{k \in N(i)} w_{ik} \frac{w_{kj}}{d_k}$$

2. Newly arrived vertices

이전 작업들과 같음.

Minimize either one of the following objective functions

**new node가 이전 노드들과 연결이 없을 경우는 아직 고려되지 않음.

$$- \sum_{j \in N(i)} w_{ji} \log p_1(v_j, v_i), \text{ or } - \sum_{j \in N(i)} w_{ji} \log p_2(v_j | v_i), \quad (10)$$

Experiment

Experiment

Data Sets

1. Language Network: word co-occurrence network from entire set of English WIKIPEDIA pages
2. Social Network
 - a. Flickr Network – more denser than Youtube Network
 - b. Youtube Network
3. Citation Networks
 - a. DBLP (Author Citation) Network
 - b. DBLP (Paper Citation) Network

Experiment

Compared Algorithms

1. Graph factorization (GF – MF techniques for graph)
2. DeepWalk
3. LINE-SGD (the weights of the edges are directly multiplied into the gradients)
4. LINE(1st)
5. LINE(2nd)
6. LINE(1st, 2nd)

Experiment

Result

Language Network – word analogy

Table 2: Results of word analogy on WIKIPEDIA data.

Algorithm	Semantic (%)	Syntactic (%)	Overall (%)	Running time
GF	61.38	44.08	51.93	2.96h
DeepWalk	50.79	37.70	43.65	16.64h
SkipGram	69.14	57.94	63.02	2.82h
LINE-SGD(1st)	9.72	7.48	8.50	3.83h
LINE-SGD(2nd)	20.42	9.56	14.49	3.94h
LINE(1st)	58.08	49.42	53.35	2.44h
LINE(2nd)	73.79	59.72	66.10	2.55h

SkipGram보다 더 높은 성능을 보임.

Word Analogy

$$\text{i.e., } d^* = \operatorname{argmax}_d \cos((\vec{u}_b - \vec{u}_a + \vec{u}_c), \vec{u}_d)$$

a : b -> c : ? 의 관계에서 ? 맞추기

Experiment

Result

Language Network – Document Classification

Table 3: Results of Wikipedia page classification on WIKIPEDIA data set.

Metric	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	GF	79.63	80.51	80.94	81.18	81.38	81.54	81.63	81.71	81.78
	DeepWalk	78.89	79.92	80.41	80.69	80.92	81.08	81.21	81.35	81.42
	SkipGram	79.84	80.82	81.28	81.57	81.71	81.87	81.98	82.05	82.09
	LINE-SGD(1st)	76.03	77.05	77.57	77.85	78.08	78.25	78.39	78.44	78.49
	LINE-SGD(2nd)	74.68	76.53	77.54	78.18	78.63	78.96	79.19	79.40	79.57
	LINE(1st)	79.67	80.55	80.94	81.24	81.40	81.52	81.61	81.69	81.67
	LINE(2nd)	79.93	80.90	81.31	81.63	81.80	81.91	82.00	82.11	82.17
	LINE(1st+2nd)	81.04**	82.08**	82.58**	82.93**	83.16**	83.37**	83.52**	83.63**	83.74**
Macro-F1	GF	79.49	80.39	80.82	81.08	81.26	81.40	81.52	81.61	81.68
	DeepWalk	78.78	79.78	80.30	80.56	80.82	80.97	81.11	81.24	81.32
	SkipGram	79.74	80.71	81.15	81.46	81.63	81.78	81.88	81.98	82.01
	LINE-SGD(1st)	75.85	76.90	77.40	77.71	77.94	78.12	78.24	78.29	78.36
	LINE-SGD(2nd)	74.70	76.45	77.43	78.09	78.53	78.83	79.08	79.29	79.46
	LINE(1st)	79.54	80.44	80.82	81.13	81.29	81.43	81.51	81.60	81.59
	LINE(2nd)	79.82	80.81	81.22	81.52	81.71	81.82	81.92	82.00	82.07
	LINE(1st+2nd)	80.94**	81.99**	82.49**	82.83**	83.07**	83.29**	83.42**	83.55**	83.66**

Significantly outperforms GF at the: ** 0.01 and * 0.05 level, paired t-test.

Experiment

Result

Language Network – Document Classification

Table 4: Comparison of most similar words using 1st-order and 2nd-order proximity.

Word	Similarity	Top similar words
good	1st	luck bad faith assume nice
	2nd	decent bad excellent lousy reasonable
information	1st	provide provides detailed facts verifiable
	2nd	infomation informaiton informations nonspammy animecons
graph	1st	graphs algebraic finite symmetric topology
	2nd	graphs subgraph matroid hypergraph undirected
learn	1st	teach learned inform educate how
	2nd	learned teach relearn learnt understand

1st : 의미론적으로 비슷한 단어

2nd : 구문론적으로 비슷한 단어

Experiment

Result

Social Network

Sparse, low avg degree

1st proximity가 보다 중요. (edge가 보다 적기에 2nd가 보다 부정확해지는 결과)

Table 5: Results of multi-label classification on the FLICKR network.

Metric	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	GF	53.23	53.68	53.98	54.14	54.32	54.38	54.43	54.50	54.48
	DeepWalk	60.38	60.77	60.90	61.05	61.13	61.18	61.19	61.29	61.22
	DeepWalk(256dim)	60.41	61.09	61.35	61.52	61.69	61.76	61.80	61.91	61.83
	LINE(1st)	63.27	63.69	63.82	63.92	63.96	64.03	64.06	64.17	64.10
	LINE(2nd)	62.83	63.24	63.34	63.44	63.55	63.55	63.59	63.66	63.69
	LINE(1st+2nd)	63.20**	63.97**	64.25**	64.39**	64.53**	64.55**	64.61**	64.75**	64.74**
Macro-F1	GF	48.66	48.73	48.84	48.91	49.03	49.03	49.07	49.08	49.02
	DeepWalk	58.60	58.93	59.04	59.18	59.26	59.29	59.28	59.39	59.30
	DeepWalk(256dim)	59.00	59.59	59.80	59.94	60.09	60.17	60.18	60.27	60.18
	LINE(1st)	62.14	62.53	62.64	62.74	62.78	62.82	62.86	62.96	62.89
	LINE(2nd)	61.46	61.82	61.92	62.02	62.13	62.12	62.17	62.23	62.25
	LINE(1st+2nd)	62.23**	62.95**	63.20**	63.35**	63.48**	63.48**	63.55**	63.69**	63.68**

Significantly outperforms DeepWalk at the: ** 0.01 and * 0.05 level, paired t-test.

Experiment

Result

Social Network

Sparse, low avg degree

1st proximity가 보다 중요. (edge가 보다 적기에 2nd가 보다 부정확해지는 결과)

괄호 내부 값은 2nd neighbors를 추가한 결과 -> 성능 향상

Table 6: Results of multi-label classification on the YOUTUBE network. The results in the brackets are on the reconstructed network, which adds second-order neighbors (i.e., neighbors of neighbors) as neighbors for vertices with a low degree.

Metric	Algorithm	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1	GF	25.43 (24.97)	26.16 (26.48)	26.60 (27.25)	26.91 (27.87)	27.32 (28.31)	27.61 (28.68)	27.88 (29.01)	28.13 (29.21)	28.30 (29.36)	28.51 (29.63)
	DeepWalk	39.68	41.78	42.78	43.55	43.96	44.31	44.61	44.89	45.06	45.23
	DeepWalk(256dim)	39.94	42.17	43.19	44.05	44.47	44.84	45.17	45.43	45.65	45.81
	LINE(1st)	35.43 (36.47)	38.08 (38.87)	39.33 (40.01)	40.21 (40.85)	40.77 (41.33)	41.24 (41.73)	41.53 (42.05)	41.89 (42.34)	42.07 (42.57)	42.21 (42.73)
	LINE(2nd)	32.98 (36.78)	36.70 (40.37)	38.93 (42.10)	40.26 (43.25)	41.08 (43.90)	41.79 (44.44)	42.28 (44.83)	42.70 (45.18)	43.04 (45.50)	43.34 (45.67)
	LINE(1st+2nd)	39.01* (40.20)	41.89 (42.70)	43.14 (43.94**)	44.04 (44.71**)	44.62 (45.19**)	45.06 (45.55**)	45.34 (45.87**)	45.69** (46.15**)	45.91** (46.33**)	46.08** (46.43**)
Macro-F1	GF	7.38 (11.01)	8.44 (13.55)	9.35 (14.93)	9.80 (15.90)	10.38 (16.45)	10.79 (16.93)	11.21 (17.38)	11.55 (17.64)	11.81 (17.80)	12.08 (18.09)
	DeepWalk	28.39	30.96	32.28	33.43	33.92	34.32	34.83	35.27	35.54	35.86
	DeepWalk (256dim)	28.95	31.79	33.16	34.42	34.93	35.44	35.99	36.41	36.78	37.11
	LINE(1st)	28.74 (29.40)	31.24 (31.75)	32.26 (32.74)	33.05 (33.41)	33.30 (33.70)	33.60 (33.99)	33.86 (34.26)	34.18 (34.52)	34.33 (34.77)	34.44 (34.92)
	LINE(2nd)	17.06 (22.18)	21.73 (27.25)	25.28 (29.87)	27.36 (31.88)	28.50 (32.86)	29.59 (33.73)	30.43 (34.50)	31.14 (35.15)	31.81 (35.76)	32.32 (36.19)
	LINE(1st+2nd)	29.85 (29.24)	31.93 (33.16**)	33.96 (35.08**)	35.46** (36.45**)	36.25** (37.14**)	36.90** (37.69**)	37.48** (38.30**)	38.10** (38.80**)	38.46** (39.15**)	38.82** (39.40**)

Significantly outperforms DeepWalk at the: ** 0.01 and * 0.05 level, paired t-test.

Experiment

Result

Citation Network

Author가 어디에 투고했는지, paper가 어디에 투고되었는지 classification

Direct graph -> rid 1st proximity

괄호는 이전과 마찬가지로, sparse -> DeepWalk outperforms 2nd

Table 7: Results of multi-label classification on DBLP(AUTHORCITATION) network.

Metric	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	DeepWalk	63.98	64.51	64.75	64.81	64.92	64.99	64.99	65.00	64.90
	LINE-SGD(2nd)	56.64	58.95	59.89	60.20	60.44	60.61	60.58	60.73	60.59
	LINE(2nd)	62.49	63.30	63.63	63.77	63.84	63.94	63.96	64.00	63.77
		(64.69*)	(65.47**)	(65.85**)	(66.04**)	(66.19**)	(66.25**)	(66.30**)	(66.12**)	(66.05**)
Macro-F1	DeepWalk	63.02	63.60	63.84	63.90	63.98	64.06	64.09	64.11	64.05
	LINE-SGD(2nd)	55.24	57.63	58.56	58.82	59.11	59.27	59.28	59.46	59.37
	LINE(2nd)	61.43	62.38	62.73	62.87	62.93	63.05	63.07	63.13	62.95
		(63.49*)	(64.42**)	(64.84**)	(65.05**)	(65.19**)	(65.26**)	(65.29**)	(65.14**)	(65.14**)

Significantly outperforms DeepWalk at the: ** 0.01 and * 0.05 level, paired t-test.

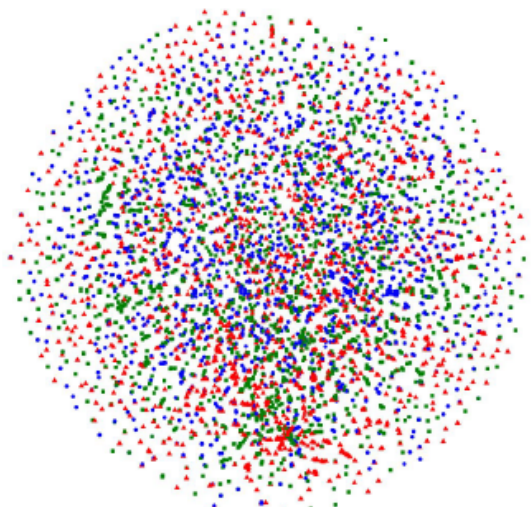
Table 8: Results of multi-label classification on DBLP(PAPERCITATION) network.

Metric	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	DeepWalk	52.83	53.80	54.24	54.75	55.07	55.13	55.48	55.42	55.90
	LINE(2nd)	58.42	59.58	60.29	60.78	60.94	61.20	61.39	61.39	61.79
		(60.10**)	(61.06**)	(61.46**)	(61.73**)	(61.85**)	(62.10**)	(62.21**)	(62.25**)	(62.80**)
Macro-F1	DeepWalk	43.74	44.85	45.34	45.85	46.20	46.25	46.51	46.36	46.73
	LINE(2nd)	48.74	50.10	50.84	51.31	51.61	51.77	51.94	51.89	52.16
		(50.22**)	(51.41**)	(51.92**)	(52.20**)	(52.40**)	(52.59**)	(52.78**)	(52.70**)	(53.02**)

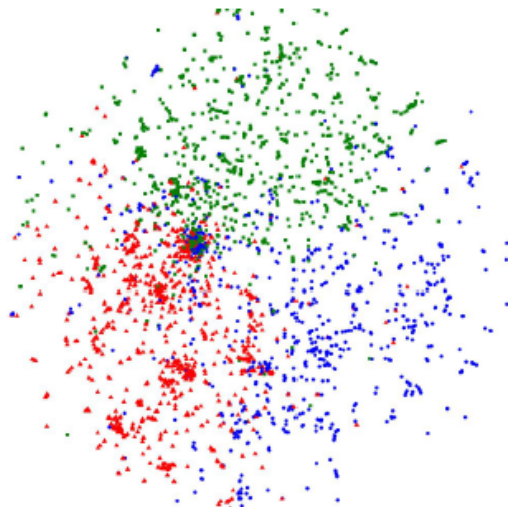
Significantly outperforms DeepWalk at the: ** 0.01 and * 0.05 level, paired t-test.

Experiment

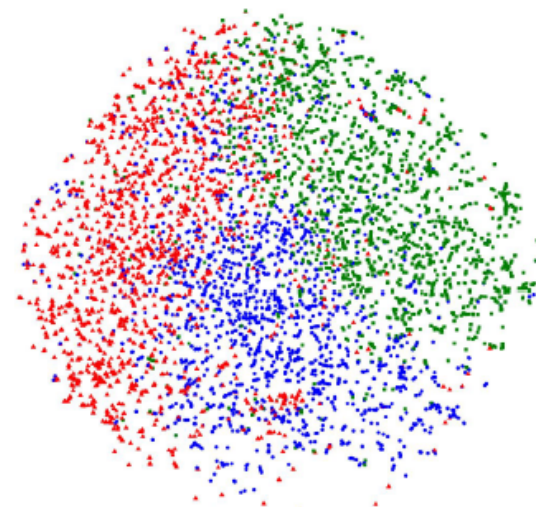
Result - visualization



(a) GF



(b) DeepWalk



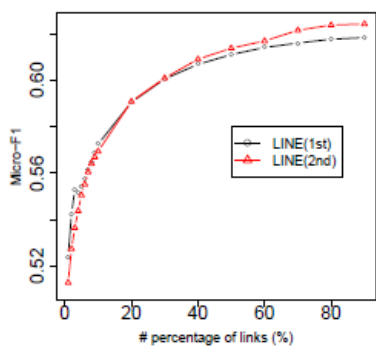
(c) LINE(2nd)

Figure 2: Visualization of the co-author network. The authors are mapped to the 2-D space using the t-SNE package with learned embeddings as input. Color of a node indicates the community of the author. Red: “data Mining,” blue: “machine learning,” green: “computer vision.”

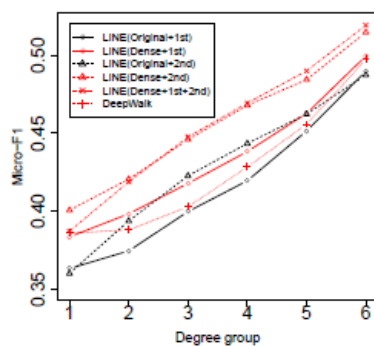
Experiment

Result

performance with respect to network sparsity, parameter sensitivity, scalability

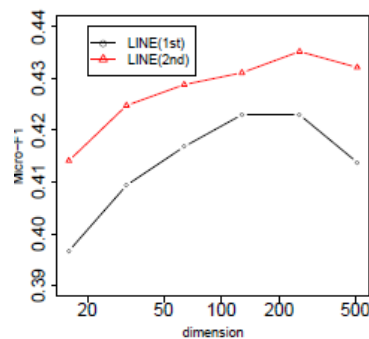


(a) Sparsity.

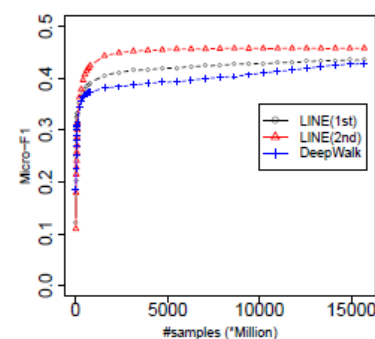


(b) Degree of vertex.

Figure 3: Performance w.r.t. network sparsity.

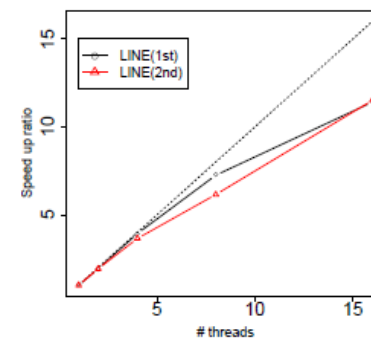


(a) #Dimension.

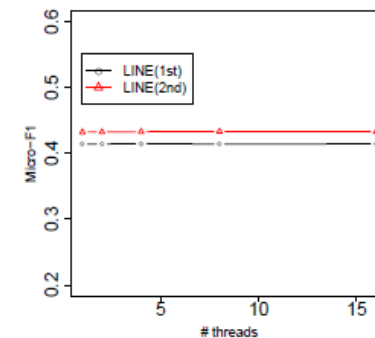


(b) #Samples.

Figure 4: Sensitivity w.r.t. dimension and samples.



(a) Speed up v.s. #threads.



(b) Micro-F1 v.s. #threads.

Figure 5: Performance w.r.t. # threads.

Conclusion

LINE 논문 요약

1st proximity & 2nd proximity를 모두 보존하는 objective function 설계.

Computational cost를 위한 edge sampling 제안, effectiveness를 보존하며 ASGD의 문제점(gradient의 가중치비례) 해결

향후 계획

4.1.3 combining 1st and 2nd proximities (jointly train the objective function)

4.3 기존 노드와 연결이 없는 새 노드가 제시되었을 때 추가연구

2nd proximity 를 넘어선 higher-order proximity 조사.

embedding of heterogeneous information networks 조사.

Thanks