Exploiting Graph Embeddings for Graph Analysis Tasks

UNIVERSITÄT PASSAU

Fatemeh Salehi Rizi

Graph Embedding Day
University of Lyon

September 7, 2018

Outline

Circle Prediction

Social labels in an ego-network

Semantic Content of Vector Embeddings

Network Centrality Measures

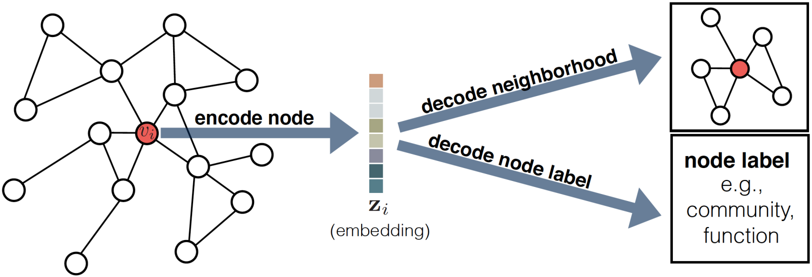
Shortest Path Approximation

Shortest path in scale-free networks

Futurework

Outline

Graph Embedding

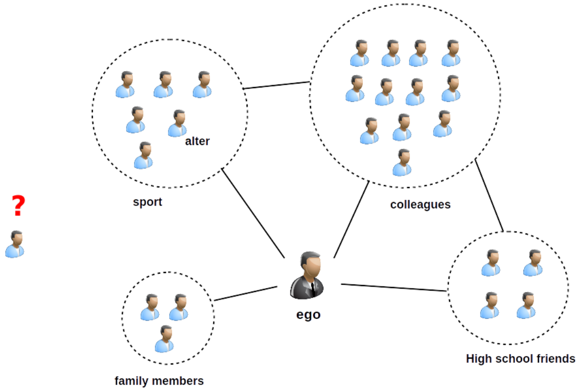


$$\text{ENC} : V \rightarrow R^d$$

$$\text{DEC} : R^d \times R^d \rightarrow R^+$$

Circle Prediction

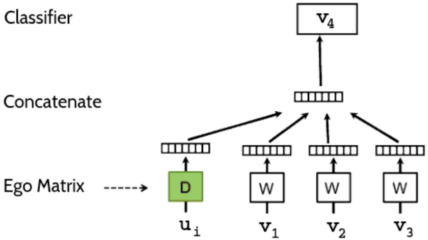
- Predicting the social circle for a new added alter to the ego-network [1]



[1] 28th International Conference on Database and Expert Systems Applications (DEXA), 2017

Circle Prediction

- node2vec for leaning global representations for all nodes glo(v)
- Walking locally over an ego-network to generate sequence of nodes
- Paragraph Vector [2] to learn local representation loc(u)

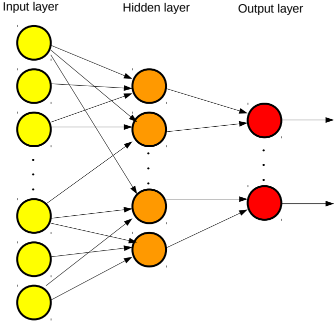


Circle Prediction

- node2vec for leaning global representations for all nodes glo(v)
- Walking locally over an ego-network to generate sequence of nodes
- Paragraph Vector [2] to learn local representation loc(u)

- Predicting circle for the alter v

- Input: loc(u) $\oplus$ glo(v)
- Profile similarity: sim(u, v)
- loc(u) $\oplus$ glo(v) $\oplus$ sim(u, v)



Circle Prediction

- Statistics of social network datasets

		Facebook	Twitter	Google+
nodes	$\lvert V \rvert$	4,039	81,306	107,614
edges	$\lvert E \rvert$	88,234	1,768,149	13,673,453
egos	$\lvert U \rvert$	10	973	132
circles	$\lvert \mathcal{C} \rvert$	46	100	468
features	f	576	2,271	4,122

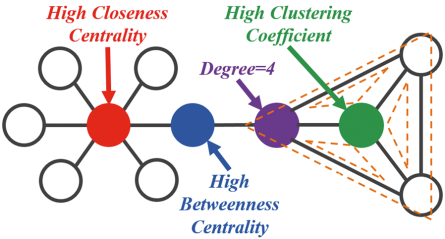
- Performance of the prediction measured by F_1-score

Approach	Facebook	Twitter	Google+
glo $\oplus$ glo	0.37	0.46	0.49
loc $\oplus$ glo	**0.42**	**0.50**	**0.52**
glo $\oplus$ glo $\oplus$ sim	0.40	0.49	0.51
loc $\oplus$ glo $\oplus$ sim	**0.45**	**0.53**	**0.55**
McAuley & Leskovec [1]	0.38	0.54	0.59

Outline

Do embeddings retain network centralities? [2]

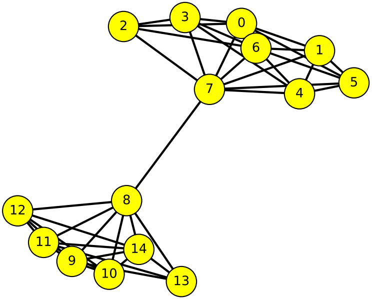
- Degree centrality $DC(u) = \deg(u)$
- Closeness centrality $CC(u) = \frac{1}{\sum_{v \in V} d(u,v)}$
- Betweenness centrality $BC(u) = \sum_{s \neq u \neq t} \frac{\sigma_{s,t}(u)}{\sigma_{s,t}}$
- Eigenvector centrality $EC(u_i) = \frac{1}{\lambda} \sum_{j=1}^{n} A_{i,j} EC(v_j)$

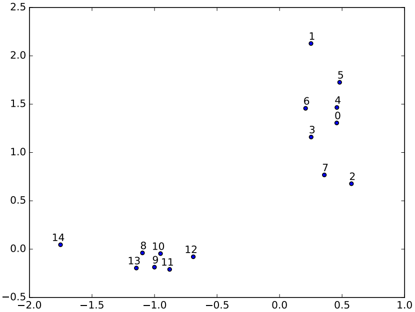


[2] Properties of Vector Embeddings in Social Networks, Algorithms Journal, 2017

Relating Embeddings and Centralities

- A pair (v_i, v_j) are similar if:
  - embedding vectors are close
  - similar network characteristics





Relating Embeddings and Centralities

- A pair (v_i, v_j) are similar if:
  - embedding vectors are close
  - similar network characteristics
- Relation

$$f(Y_i, Y_j) \sim \sum_{i=1}^{k} w_i \, \mathrm{sim}(v_i, v_j)$$

  - Y_i is the embedding vector of v_i
  - w_i is the weight of the centrality i
  - p_i is a function computes similarity
  - k is the number of centrality measures

Relating Embeddings and Centralities

- A pair (v_i, v_j) are similar if:
  - embedding vectors are close
  - similar network characteristics
- Relation

$$f(Y_i, Y_j) \sim \sum_{i=1}^{k} w_i \,\mathrm{sim}(v_i, v_j)$$

  - Y_i is the embedding vector of v_i
  - w_i is the weight of the centrality i
  - p_i is a function computes similarity
  - k is the number of centrality measures

Learning to Rank can learn weights

Learning to Rank

- Ranking nodes according similarity in the embedding space
- Feature matrix according similarity in the network
- rankSVM objective function:

$$\frac{1}{2}w^T w + C \sum_{(i,j)\in V} \max(0, 1 - w^T(x_i - x_j))$$

$$w = (w_{DC}, w_{CC}, w_{BC}, w_{EC})$$

Learning to Rank

- Every pair (v_i, v_j) has a centrality similarity
  - P_{v_i}: histogram of centrality distribution in $N(v_i)$
  - Q_{v_j}: histogram of centrality distribution in $N(v_j)$
  - centrality similarity: $1 - D_{KL}(P_{v_i} \| Q_{v_j})$
- Feature matrix $X \in \mathbb{R}^{|z| \times 4}$, $z = n \times (n-1)$

$$X = \begin{bmatrix} \text{sim}_{\text{DC}}(v_1, v_2) & \text{sim}_{\text{CC}}(v_1, v_2) & \text{sim}_{\text{BC}}(v_1, v_2) & \text{sim}_{\text{EC}}(v_1, v_2) \\ \text{sim}_{\text{DC}}(v_1, v_3) & \text{sim}_{\text{CC}}(v_1, v_3) & \text{sim}_{\text{BC}}(v_1, v_3) & \text{sim}_{\text{EC}}(v_1, v_3) \\ \text{sim}_{\text{DC}}(v_1, v_4) & \text{sim}_{\text{CC}}(v_1, v_4) & \text{sim}_{\text{BC}}(v_1, v_4) & \text{sim}_{\text{EC}}(v_1, v_4) \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Learning to Rank

- Every node v_i sort all other nodes according to $Y_i \cdot Y_j$
- $v_i : [v_1, v_2, \cdots, v_{n-1}]$
- Every pair (v_i, v_j) has a rank label
- Ground-truth $y \in \mathbb{R}^{|z| \times 1}$, $z = n \times (n-1)$

$$y = \begin{bmatrix} \text{rank}(v_1, v_2) \\ \text{rank}(v_1, v_3) \\ \text{rank}(v_1, v_4) \\ \vdots \end{bmatrix}$$

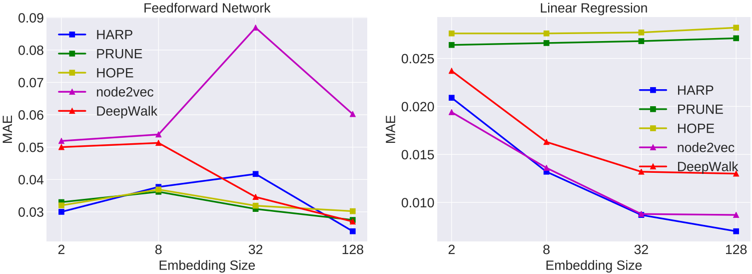
Semantic content of embeddings

- Deepwalk: d=128, k=5, r=10, l=80
- node2vec: d=128, q=5, p=0.1
- line: d=128

Dataset	Weight	DeepWalk	LINE	node2vec
Facebook	w_{DC}	0.09± 0.02	-0.15 ±0.05	**0.82±0.01**
	w_{CC}	-0.01± 0.04	-0.07 ±0.00	0.04±0.00
	w_{BC}	**0.64± 0.03**	**-0.55±0.07**	-0.01±0.04
	w_{EC}	**-0.64±0.02**	**-0.68±0.08**	-0.07±0.00
Twitter	w_{DC}	0.07±0.09	-0.09 ±0.05	**0.53±0.01**
	w_{CC}	-0.15 ±0.00	-0.00 ±0.08	0.04 ±0.17
	w_{BC}	**0.51±0.04**	**-0.69±0.00**	-0.11 ±0.10
	w_{EC}	**-0.71±0.05**	**-0.58±0.01**	-0.03 ±0.01
Google+	w_{DC}	0.02±0.04	-0.00 ±0.10	**0.65±0.00**
	w_{CC}	-0.05 ±0.11	-0.04 ±0.09	0.09 ±0.07
	w_{BC}	**0.55±0.05**	**-0.53±0.07**	-0.14 ±0.00
	w_{EC}	**-0.63±0.03**	**-0.68±0.06**	-0.07 ±0.03

Predicting Centrality Values

Dataset	$|V|$	Average Closeness	std
Facebook	$4,039$	0.2759	0.0349



Linear Regression gives the minimum MAE by HARP: 0.0070

Outline

Shortest-path Problem

Single-Source Shortest-Path (SSSP)

Given a Graph $G = (V, E)$ and Source $s \in V$, compute all distances $\delta(s, v)$, where $v \in V$.

All-Pairs Shortest-Path (APSP)

Given a graph $G = (V, E)$, compute all distances between a source vertex s and a destination v, where s and v are elements of the set V.

Shortest-path Problem

Single-Source Shortest-Path (SSSP)

Given a Graph $G = (V, E)$ and Source $s \in V$, compute all distances $\delta(s, v)$, where $v \in V$.

All-Pairs Shortest-Path (APSP)

Given a graph $G = (V, E)$, compute all distances between a source vertex s and a destination v, where s and v are elements of the set V.

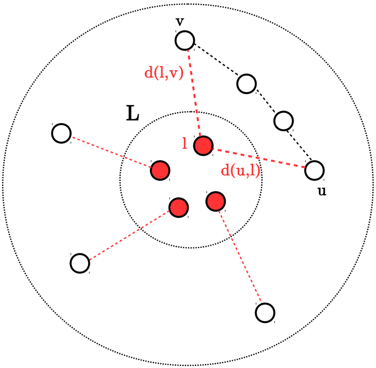
- Exact methods: Algorithms try to find the exact shortest-paths between vertices in any type of graphs
- Approximation Methods: Algorithms attempt to compute shortest-paths between nodes by querying only some of the distances.

Exact Methods

Algorithm	Time Complexity
Dijkstra (V times) [14]	$O(\lvert V\rvert^2 \log \lvert V\rvert + \lvert V\rvert\lvert E\rvert \log \lvert V\rvert)$
Floyd-Warshall [3]	$O(\lvert V\rvert^3)$
Thorup [4]	$O(\lvert E\rvert\lvert V\rvert)$
Pettie & Ramachandran [5]	$O(\lvert E\rvert\lvert V\rvert \log \alpha(\lvert E\rvert, \lvert V\rvert))$
Williams [6]	$O(\lvert V\rvert^3/2^{\Omega(\log \lvert V\rvert)^{1/2}})$
Han and Takaoka [15]	$O(\lvert V\rvert^3(\log\log \lvert V\rvert)/(\log \lvert V\rvert)^2)$
Fredman [16]	$O(\lvert V\rvert^3(\log\log \lvert V\rvert)/\log \lvert V\rvert 1/3)$
T. M. Chan [17]	$O(\lvert V\rvert^3/\log \lvert V\rvert)$

Approximation Methods

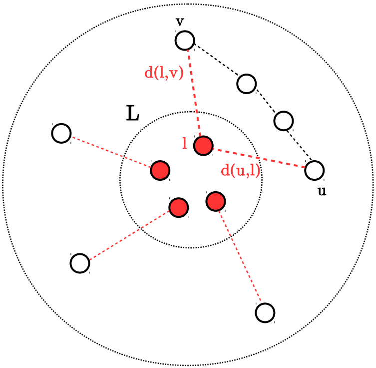
- Landmark-based Methods [8, 9, 10, 11]
- A subset L of vertices as landmarks
  - $k = |L|$, $k \ll |V|$



Approximation Methods

- Landmark-based Methods [8, 9, 10, 11]

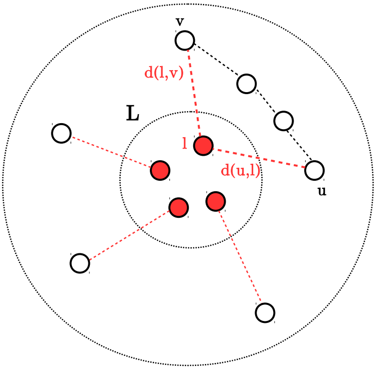
- A subset L of vertices as landmarks
  - $k = |L|$, $k \ll |V|$
- For all $l \in L$ and $u \in V$: $d(l, u)$
  - BFS: $O(k(|E| + |V|))$



Approximation Methods

- Landmark-based Methods [8, 9, 10, 11]

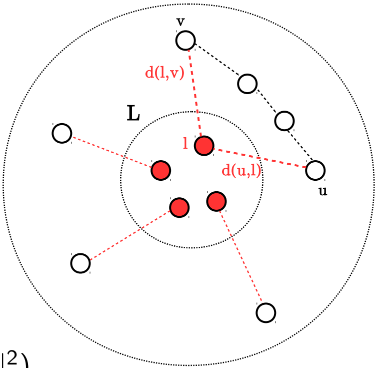
- A subset L of vertices as landmarks
  - $k = |L|$, $k \ll |V|$
- For all $l \in L$ and $u \in V$: $d(l, u)$
  - BFS: $O(k(|E| + |V|))$
- $d(u, v) = \min(d(u, l) + d(l, v))$
  - Query time: $O(k)$



Approximation Methods

- Landmark-based Methods [8, 9, 10, 11]

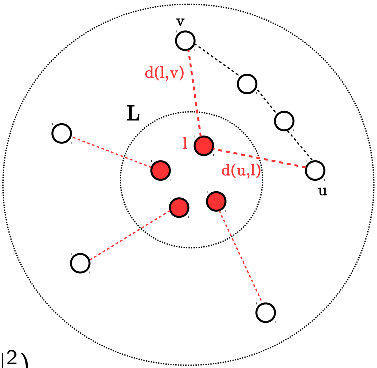
- A subset L of vertices as landmarks
  - $k = |L|$, $k \ll |V|$
- For all $l \in L$ and $u \in V$: $d(l, u)$
  - BFS: $O(k(|E| + |V|))$
- $d(u, v) = \min(d(u, l) + d(l, v))$
  - Query time: $O(k)$
- For all pairs: $O(k(|E| + |V|)) + O(k|V|^2)$



Approximation Methods

- Landmark-based Methods [8, 9, 10, 11]

- A subset L of vertices as landmarks
  - $k = |L|$, $k \ll |V|$
- For all $l \in L$ and $u \in V$: $d(l, u)$
  - BFS: $O(k(|E| + |V|))$
- $d(u, v) = \min(d(u, l) + d(l, v))$
  - Query time: $O(k)$
- For all pairs: $O(k(|E| + |V|)) + O(k|V|^2)$



Optimal Landmark selection is a NP-hard problem!

Our Approach [3]

Algorithm 1: All-Pairs Shortest Path Approximation

```
Data: graph G = (V, E)
for u, v ∈ V do
    if v ∈ N_u or u ∈ N_v then
        return 1
    else
        return SP(u, v)
```

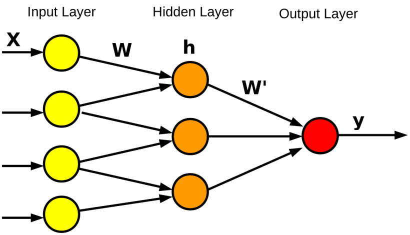
- N_u is a set of u's direct neighbors
- SP is a neural network approximation function

[3] IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2018

Approximator

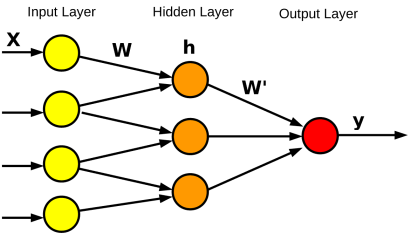
- A Feedforward Network
- Mapping function: $R^d \to R^+$

- Input layer
  - Hadamard $\odot$
  - Average $\oslash$
  - Concatenation $\oplus$
  - Subtraction $\ominus$

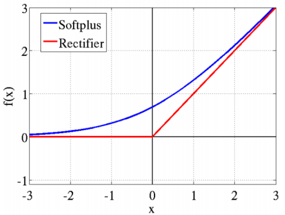


Approximator

- A Feedforward Network
- Mapping function: $R^d \rightarrow R^+$



- Input layer
  - Hadamard $\odot$
  - Average $\oslash$
  - Concatenation $\oplus$
  - Subtraction $\ominus$
- Hidden layer: $h = \max(0, z)$, $z = xw + b$
- Output layer: $y = \ln(1 + e^{z'})$, $z' = hw' + b$

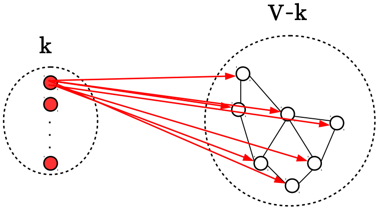


Our Approach

1. Node embeddings $O(|V|)$
   - node2vec
   - Poincaré

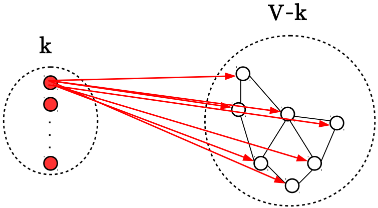
Our Approach

1. Node embeddings $O(|V|)$
   - node2vec
   - Poincaré
2. Training pairs



   - Selecting k random landmarks
   - Breadth-first search from landmarks to others to obtain the training shortest paths $O(k(|E| + |V|))$
   - $k(|V| - k)$ training pairs

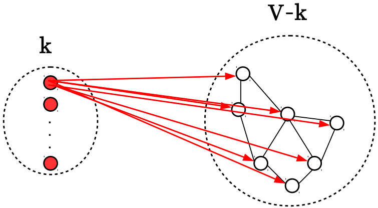
Our Approach



1. Node embeddings $O(|V|)$
   - node2vec
   - Poincaré
2. Training pairs
   - Selecting k random landmarks
   - Breadth-first search from landmarks to others to obtain the training shortest paths $O(k(|E| + |V|))$
   - $k(|V| - k)$ training pairs
3. Train a Feedforward network on training pairs $k(|V| - k)O(1)$

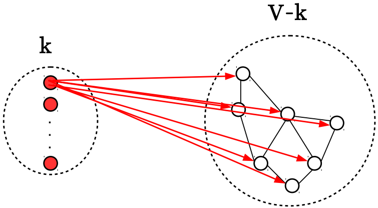
Our Approach

1. Node embeddings $O(|V|)$
   - node2vec
   - Poincaré
2. Training pairs



   - Selecting k random landmarks
   - Breadth-first search from landmarks to others to obtain the training shortest paths $O(k(|E| + |V|))$
   - $k(|V| - k)$ training pairs
3. Train a Feedforward network on training pairs $k(|V| - k)O(1)$
4. Test the network on remaining pairs (unseen pairs)

Our Approach



1. Node embeddings $O(|V|)$
   - node2vec
   - Poincaré
2. Training pairs
   - Selecting k random landmarks
   - Breadth-first search from landmarks to others to obtain the training shortest paths $O(k(|E| + |V|))$
   - $k(|V| - k)$ training pairs
3. Train a Feedforward network on training pairs $k(|V| - k)O(1)$
4. Test the network on remaining pairs (unseen pairs)

The total run time:

$$O(|V|) + kO(|E| + |V|) + k(|V| - k)O(1) + C < O(k|V||E|)$$

Approximation Quality

- Error Estimation
  - Mean Absolute Error (MAE): $\frac{1}{n_t} \sum |d - \hat{d}|$
  - Mean Relative Error (MRE): $\frac{1}{n_t} \sum \frac{|d - \hat{d}|}{d}$
- Test and Train pairs

Dataset	$\|V\|$	$\|E\|$	$\overline{d}$	Training pairs	Test pairs
Facebook	$4,039$	$88,234$	4.32	$1,022,640$	$109,978$
Blog Catalog	$88,784$	4186390	2.72	$1,409,700$	$88,316$
Youtube	$1,134,890$	$2,987,624$	5.5	$2,452,757$	$184,413$
Flickr	$1,715,255$	$15,551,250$	5.13	$2,579,437$	$112,967$

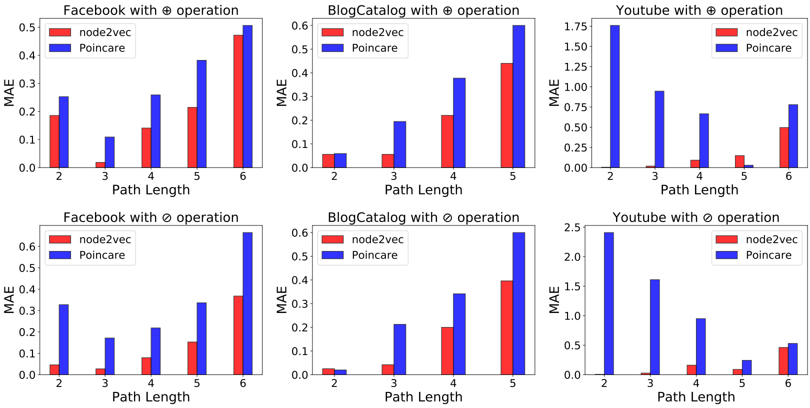
- Facebook
  - 30 sec (node2vec) + 5 min (gather pairs) + 3 min (training and test)

Error Estimation

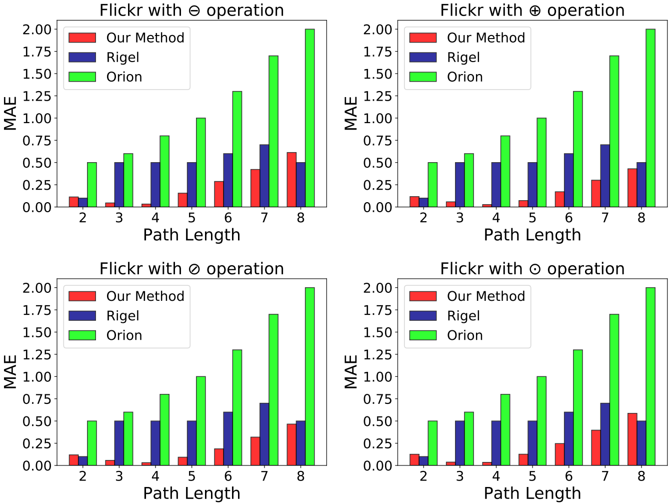
- Feedforward Neural Network

Dataset	Embedding	Size	MAE				MRE			
			$\ominus$	$\oplus$	$\oslash$	$\odot$	$\ominus$	$\oplus$	$\oslash$	$\odot$
Facebook	node2vec	32	0.480	0.415	0.233	0.531	0.175	0.164	0.068	0.188
		128	0.197	0.258	**0.118**	0.217	0.071	0.099	**0.038**	0.081
	Poincaré	32	0.592	0.594	0.552	0.604	0.214	0.211	0.218	0.212
		128	0.437	0.315	0.372	0.608	0.169	0.115	0.142	0.246
BlogCatalog	node2vec	32	0.277	0.242	0.197	0.193	0.092	0.103	0.067	0.067
		128	0.220	0.275	0.159	**0.154**	0.077	0.119	0.064	**0.059**
	Poincaré	32	0.338	0.338	0.343	0.338	0.108	0.108	0.112	0.108
		128	0.331	0.354	0.277	0.338	0.115	0.138	0.097	0.108
Youtube	node2vec	32	0.676	0.265	0.455	0.625	0.230	0.066	0.163	0.223
		128	0.344	**0.154**	0.174	0.244	0.101	**0.034**	0.040	0.061
	Poincaré	32	1.095	0.708	1.134	0.774	0.429	0.264	0.446	0.291
		128	1.270	1.185	1.746	0.771	0.497	0.468	0.681	0.262
Flickr	node2vec	32	0.699	0.295	0.564	0.525	0.250	0.086	0.183	0.198
		128	0.238	**0.168**	0.181	0.222	0.171	**0.074**	0.178	0.179
	Poincaré	32	0.995	0.808	1.022	0.874	0.349	0.284	0.429	0.278
		128	0.803	0.662	0.807	0.764	0.397	0.432	0.566	0.364

Error Distribution



Comparing to State-of-the-art



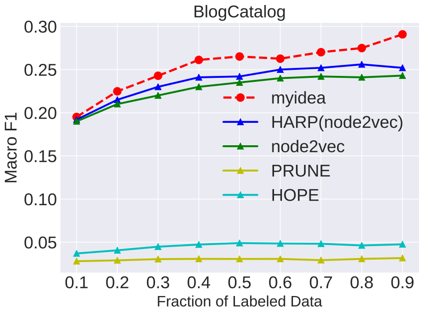
Outline

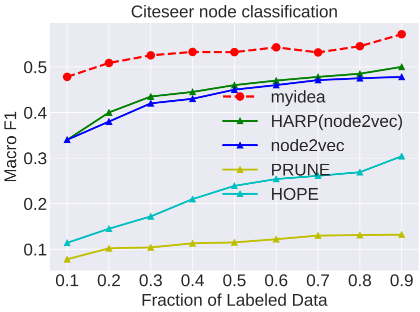
Future work

- For future:
  - Approximating longer distances among nodes
  - Learning embeddings which retain centralities

Future work

- For future:
  - Approximating longer distances among nodes
  - Learning embeddings which retain centralities
  - An idea of graph embedding





References (1)

McAuley, Julian, and Jure Leskovec. Discovering social circles in ego networks. ACM Transactions on Knowledge Discovery from Data (TKDD) 8, no. 1, pp.4, 2014.

Le, Quoc V., and Tomas Mikolov. Distributed Representations of Sentences and Documents. In ICML, vol. 14, pp. 1188-1196. 2014.

Floyd, Robert W. (June 1962). "Algorithm 97: Shortest Path". Communications of the ACM. 5 (6): 345. doi:10.1145/367766.368168

Thorup, Mikkel (1999). "Undirected single-source shortest paths with positive integer weights in linear time". Journal of the ACM. 46 (3): 362394. doi:10.1145/316542.316548. Retrieved 28 November 2014.

Pettie, Seth; Ramachandran, Vijaya (2002). Computing shortest paths with comparisons and additions. Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms. pp. 267276. ISBN 0-89871-513-X.

Williams, Ryan (2014). "Faster all-pairs shortest paths via circuit complexity". Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC '14). New York: ACM. pp. 664673. arXiv:1312.6680Freely accessible. doi:10.1145/2591796.2591811. MR 3238994.

Hamilton, William L., Rex Ying, and Jure Leskovec. "Representation Learning on Graphs: Methods and Applications." arXiv preprint arXiv:1709.05584 (2017).

References (2)

Tretyakov, Konstantin, Abel Armas-Cervantes, Luciano Garca-Bauelos, Jaak Vilo, and Marlon Dumas. "Fast fully dynamic landmark-based estimation of shortest path distances in very large graphs." In Proceedings of the 20th ACM international conference on Information and knowledge management, pp. 1785-1794. ACM, 2011.

Potamias, Michalis, Francesco Bonchi, Carlos Castillo, and Aristides Gionis. "Fast shortest path distance estimation in large networks." In Proceedings of the 18th ACM conference on Information and knowledge management, pp. 867-876. ACM, 2009.

Takes, Frank W., and Walter A. Kosters. "Adaptive landmark selection strategies for fast shortest path computation in large real-world graphs." In Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on, vol. 1, pp. 27-34. IEEE, 2014.

Akiba, Takuya, Yoichi Iwata, and Yuichi Yoshida. "Fast exact shortest-path distance queries on large networks by pruned landmark labeling." In Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, pp. 349-360. ACM, 2013.

Chen, Haochen, Bryan Perozzi, Yifan Hu, and Steven Skiena. "HARP: Hierarchical Representation Learning for Networks." arXiv preprint arXiv:1706.07845 (2017).

G. Koch, R. Zemel, and R. Salakhutdinov, Siamese neural networks for one-shot image recognition, in ICML Deep Learning Workshop, vol. 2, 2015.

Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). "Section 24.3: Dijkstra's algorithm". Introduction to Algorithms (Second ed.). MIT Press and McGrawHill. pp. 595-601. ISBN 0-262-03293-7.

References (3)

Y. Han and T. Takaoka. An o(n3 log log n/ log2 n) time algorithm for all pairs shortest paths. Proceedings of the 13th Scandinavian conference on Algorithm Theory, pages 131141, 2012.

M. Fredman. New bounds on the complexity of the shortest path problem. SIAM, pages 83-89, 1976.

T. M. Chan. All-pairs shortest paths for unweighted undirected graphs in o(mn) time. Proceed- ings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm, pages 514-523, 2006.

Thanks for your attention!