SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Oana Balalau and Sagar Goyal

Inria and Ecole Polytechnique (France) & Microsoft (Canada)

SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Subgraphs are very important structures, e.g. a cascade in an information network or a community in a social graph.

SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Subgraphs are very important structures, e.g. a cascade in an information network or a community in a social graph.

Limitation of current techniques for subgraph embeddings:

- they are supervised techniques;
- they can tackle only a specific type of subgraph.

SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Subgraphs are very important structures, e.g. a cascade in an information network or a community in a social graph.

Limitation of current techniques for subgraph embeddings:

- they are supervised techniques;
- they can tackle only a specific type of subgraph.

We present an approach for computing in an unsupervised fashion embeddings of any type of subgraph.

Problem statement. Given a directed graph $G = (V, E)$, a set of subgraphs $S_1, S_2, \cdots, S_k$ of G, compute the embeddings of the subgraphs.

SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Problem statement. Given a directed graph $G = (V, E)$, a set of subgraphs $S_1, S_2, \cdots, S_k$ of G, compute the embeddings of the subgraphs.

Solution. Compute subgraph embeddings such that the embeddings preserve an input similarity distribution between subgraphs.

SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Problem statement. Given a directed graph $G = (V, E)$, a set of subgraphs $S_1, S_2, \cdots, S_k$ of G, compute the embeddings of the subgraphs.

Solution. Compute subgraph embeddings such that the embeddings preserve an input similarity distribution between subgraphs.

Contributions:

- We define a novel subgraph to subgraph proximity measure;

SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Problem statement. Given a directed graph $G = (V, E)$, a set of subgraphs $S_1, S_2, \cdots, S_k$ of G, compute the embeddings of the subgraphs.

Solution. Compute subgraph embeddings such that the embeddings preserve an input similarity distribution between subgraphs.

Contributions:

- We define a novel subgraph to subgraph proximity measure;
- We introduce a framework that learns comprehensive subgraphs embeddings;

SubRank: Subgraph Embeddings via a Subgraph Proximity Measure

Problem statement. Given a directed graph $G = (V, E)$, a set of subgraphs $S_1, S_2, \cdots, S_k$ of G, compute the embeddings of the subgraphs.

Solution. Compute subgraph embeddings such that the embeddings preserve an input similarity distribution between subgraphs.

Contributions:

- We define a novel subgraph to subgraph proximity measure;
- We introduce a framework that learns comprehensive subgraphs embeddings;
- We validate our approach on several data mining tasks.

Subgraph Proximity Measure

Intuition: Important nodes in subgraph S_i should be close to important nodes in subgraph S_j.

Emmanuel wants to send macarons to Halimah, from France to Singapore.

(a) Emmanuel

(b) The macarons

(c) Halimah

Subgraph Proximity Measure

In a graph G, given subgraphs S_i and S_j we have:

- Emmanuel is a random surfer in the subgraph S_i,
- Halimah is a random surfer in the subgraph S_j,
- and MacaronPost is a random surfer in graph G.

Subgraph Proximity Measure

In a graph G, given subgraphs S_i and S_j we have:

- Emmanuel is a random surfer in the subgraph S_i,
- Halimah is a random surfer in the subgraph S_j,
- and MacaronPost is a random surfer in graph G.

Emmanuel decides to send macarons to Halimah via MacaronPost.

Subgraph Proximity Measure

In a graph G, given subgraphs S_i and S_j we have:

- Emmanuel is a random surfer in the subgraph S_i,
- Halimah is a random surfer in the subgraph S_j,
- and MacaronPost is a random surfer in graph G.

Emmanuel decides to send macarons to Halimah via MacaronPost.

Random walk: MacaronPost starts from the node v_i Emmanuel is visiting $(PR_{S_i}(v_i))$ and will reach a node $v_j \in S_j$ with probability $PPR(v_i, v_j)$.

Halimah will be there to receive the message with probability $PR_{S_j}(v_j)$.

Subgraph Proximity Measure

Let S_i and S_j be two subgraphs in a directed graph G. Their proximity is:

$$px(S_i, S_j) = \sum_{v_i \in S_i} PR_{S_i}(v_i) \sum_{v_j \in S_j} PR_{S_j}(v_j) \cdot PPR(v_i, v_j), \tag{1}$$

where $PR_{S_i}(v_i)$ is the PageRank of node v_i in the S_i, and $PPR(v_i, v_j)$ the PageRank of node v_j personalized for node v_i in G.

Subgraph Proximity Measure

Let S_i and S_j be two subgraphs in a directed graph G. Their proximity is:

$$px(S_i, S_j) = \sum_{v_i \in S_i} PR_{S_i}(v_i) \sum_{v_j \in S_j} PR_{S_j}(v_j) \cdot PPR(v_i, v_j), \tag{1}$$

where $PR_{S_i}(v_i)$ is the PageRank of node v_i in the S_i, and $PPR(v_i, v_j)$ the PageRank of node v_j personalized for node v_i in G.

Given $S = \{S_1, S_2, \cdots S_k\}$, the normalized proximity is:

$$\hat{px}(S_i, S_j) = \frac{px(S_i, S_j)}{\sum_{S_k \in S} px(S_i, S_k)} \tag{2}$$

SubRank: Subgraph Embedding Algorithm

Algorithm intuition: embeddings preserve the proximity distribution between subgraphs, similar to a node embedding approach [Tsitsulin et al., 2018].

SubRank: Subgraph Embedding Algorithm

Algorithm intuition: embeddings preserve the proximity distribution between subgraphs, similar to a node embedding approach [Tsitsulin et al., 2018].

We minimize the Kullback-Leibler (KL) divergence between the two distributions:

$$\sum_{S_k \in S} KL(\hat{px}_G(S_k, \cdot), sim_E(S_k, \cdot))$$

SubRank: Subgraph Embedding Algorithm

Algorithm intuition: embeddings preserve the proximity distribution between subgraphs, similar to a node embedding approach [Tsitsulin et al., 2018].

We minimize the Kullback-Leibler (KL) divergence between the two distributions:

$$\sum_{S_k \in S} KL(\hat{px}_G(S_k, \cdot), sim_E(S_k, \cdot))$$

The algorithm uses Noise Contrastive Estimation and requires sampling the subgraph proximity distribution. We efficiently sample the distribution using a Monte Carlo estimation.

Graph mining applications

We validate our embeddings on several real-world tasks:

- community detection
- link prediction
- cascade growth prediction

Graph mining applications

We validate our embeddings on several real-world tasks:

- community detection
- link prediction
- cascade growth prediction

The first two tasks will compute embeddings of ego networks (the neighborhood of a node), which we will use to represent nodes.

Graph mining applications

We validate our embeddings on several real-world tasks:

- community detection
- link prediction
- cascade growth prediction

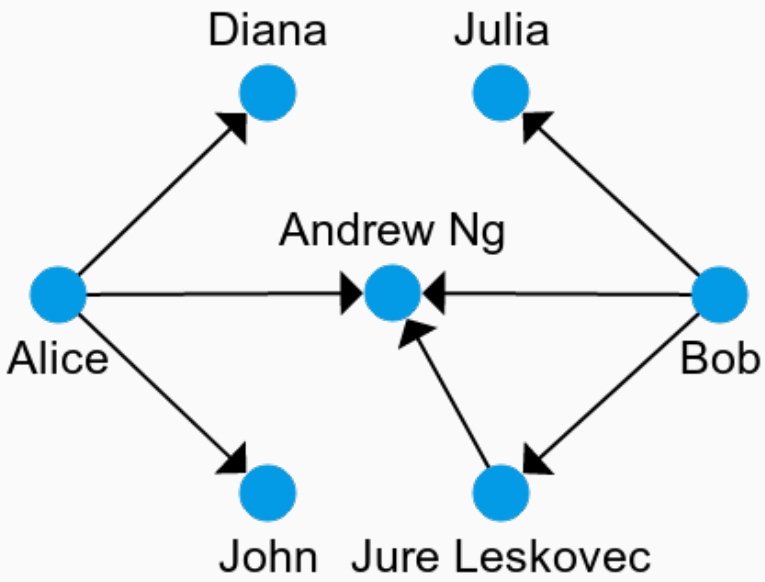
The first two tasks will compute embeddings of ego networks (the neighborhood of a node), which we will use to represent nodes.

For the last task, we will compute embeddings of cascades (trees in an information propagation graph).

Proximity of ego networks

Alice and Bob are two computer scientists using Twitter.

A path-based similarity measure between Alice and Bob will return a similarity 0, while it will return high values between Alice and Andrew Ng and Bob and Andrew Ng.



Proximity of ego networks

Experimental setting:

- Task: node clustering, node classification, link prediction

Proximity of ego networks

Experimental setting:

- Task: node clustering, node classification, link prediction

- Our method, SubRank, will represent each node using the embedding of its ego network.

Proximity of ego networks

Experimental setting:

- Task: node clustering, node classification, link prediction
- Our method, SubRank, will represent each node using the embedding of its ego network.
- We compare against node embeddings methods: DeepWalk, LINE, node2vec, VERSE, and subgraph embedding methods: sub2vec, VerseAvg.

Proximity of ego networks

Experimental setting:

- Task: node clustering, node classification, link prediction
- Our method, SubRank, will represent each node using the embedding of its ego network.
- We compare against node embeddings methods: DeepWalk, LINE, node2vec, VERSE, and subgraph embedding methods: sub2vec, VerseAvg.
- In VerseAvg, we compute a node embedding as the average of the embeddings of the nodes in its ego network, computed using VERSE.

Proximity of ego networks

Experimental setting:

- Task: node clustering, node classification, link prediction
- Our method, SubRank, will represent each node using the embedding of its ego network.
- We compare against node embeddings methods: DeepWalk, LINE, node2vec, VERSE, and subgraph embedding methods: sub2vec, VerseAvg.
- In VerseAvg, we compute a node embedding as the average of the embeddings of the nodes in its ego network, computed using VERSE.
- In sub2vec, the authors compute node embedding using their ego network. Their approach is inspired by the Paragraph2vec model.

Proximity of ego networks

Dataset	Type	$\|V\|$	$\|E\|$	$\|L\|$
Citeseer	Citation	3.3K	4.7K	6
Cora	Citation	2.7K	5.4K	7
Polblogs	Hyperlink	1.4K	19K	2
Cithep	Citation	34K	421k	1
DBLP	Co-authorship	66K	542K	20

Table 1: Dataset description: type, vertices V, edges E, node labels L.

Proximity of ego networks

	Dataset			
Method	Citeseer	Cora	Polblogs	DBLP
DeepWalk	0.015	0.018	0.013	0.314
node2vec	0.023	0.100	0.013	0.336
LINE	0.084	0.208	0.448	0.284
VERSE	0.103	0.257	0.024	0.363
VerseAvg	0.125	0.310	0.318	0.360
sub2vec	0.007	0.004	0.001	0.001
SubRank	0.179	0.347	0.021	0.357

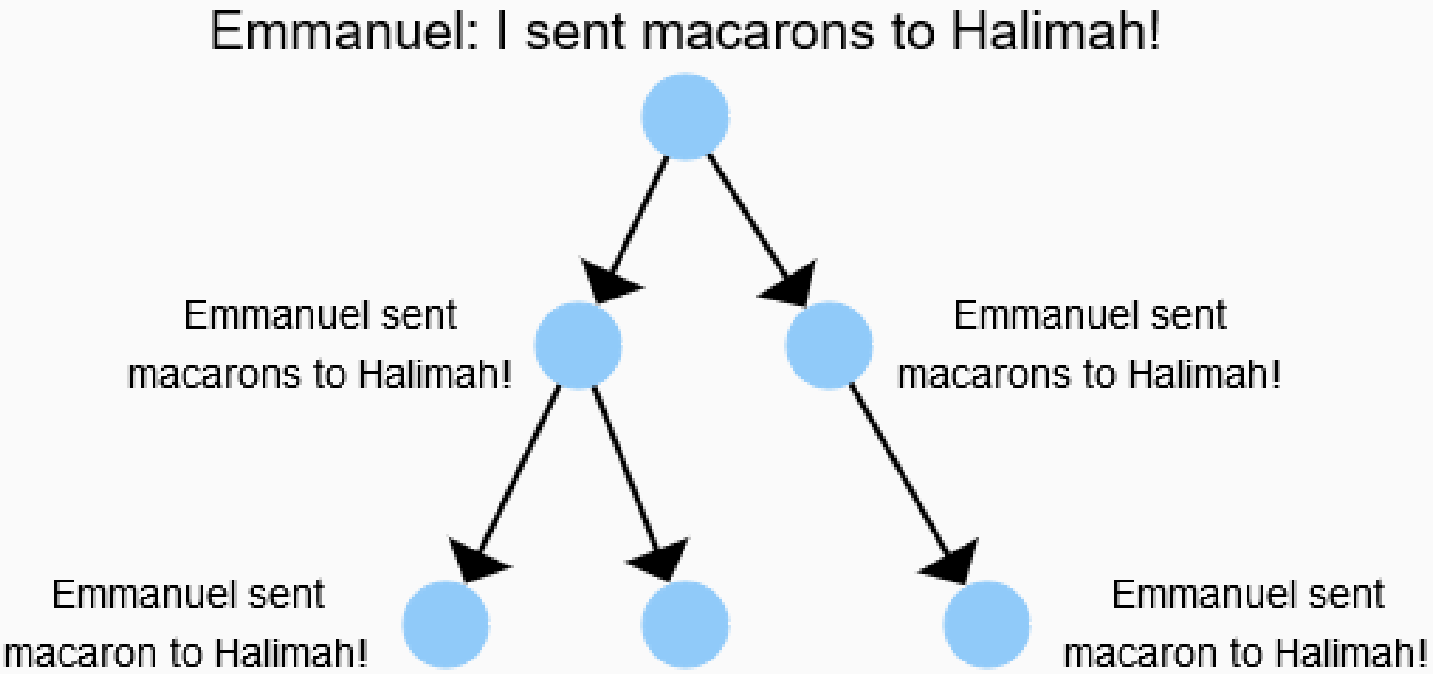
Table 2: Normalized Mutual Information (NMI) for node clustering.

Proximity of ego networks

	Dataset				
Method	Cora	Citeseer	Polblogs	Cithep	DBLP
DeepWalk	67.42	50.75	81.93	72.24	97.83
node2vec	67.04	59.54	82.76	75.58	98.56
LINE	71.49	61.48	83.72	84.85	98.26
VERSE	70.17	67.16	85.83	92.18	99.27
sub2vec	52.46	50.32	53.30	51.08	50.28
SubRank	80.10	82.10	82.70	94.10	99.30

Table 3: Accuracy for link prediction.

Proximity of cascade subgraphs



In a cascade, nodes that reshared more recently are more visible to their neighbors. In a directed tree, nodes on the same level have the same Pagerank, and the Pagerank increases as with depth.

Proximity of cascade subgraphs

Given:

- a social network $G = (V, E)$
- and a set of cascade graphs $C_i = (V_c, E_c)$ with $V_c \in V$,

two cascades have a high subgraph proximity if:

nodes joining later the cascades (i.e. on lower levels in the trees) are "close" according to their Personalized Pagerank in G.

Intuition: Defining a similarity between cascades can inform us of how the cascades will evolve, for example, their growth in a period of time.

Proximity of cascade subgraphs

Experimental setting:

- Given in input a social graph $G = (V, E)$, a set of cascades $C_i = (V_c, E_c)$ with $V_c \in V$, and a time window t. What is the size of C_i after time t?

Proximity of cascade subgraphs

Experimental setting:

- Given in input a social graph $G = (V, E)$, a set of cascades $C_i = (V_c, E_c)$ with $V_c \in V$, and a time window t. What is the size of C_i after time t?

- Our method, SubRank, computes the embedding of the cascades, and to predict the cascade growth we use an MLP regressor.

Proximity of cascade subgraphs

Experimental setting:

- Given in input a social graph $G = (V, E)$, a set of cascades $C_i = (V_c, E_c)$ with $V_c \in V$, and a time window t. What is the size of C_i after time t?
- Our method, SubRank, computes the embedding of the cascades, and to predict the cascade growth we use an MLP regressor.
- We compare against two end-to-end deep learning frameworks for cascade predictions, DeepCas and DeepHawkes.

Proximity of cascade subgraphs

Experimental setting:

- Given in input a social graph $G = (V, E)$, a set of cascades $C_i = (V_c, E_c)$ with $V_c \in V$, and a time window t. What is the size of C_i after time t?
- Our method, SubRank, computes the embedding of the cascades, and to predict the cascade growth we use an MLP regressor.
- We compare against two end-to-end deep learning frameworks for cascade predictions, DeepCas and DeepHawkes.
- In addition, we compare against a simple baseline in which a cascade is represented by the average embedding of its nodes, VerseAvg.

Proximity of cascade subgraphs

Datasets:

- AMiner is a dataset on scientific citations. A node is an author and an edge represents a citation. A cascade contains all the citations of a given paper.

Proximity of cascade subgraphs

Datasets:

- AMiner is a dataset on scientific citations. A node is an author and an edge represents a citation. A cascade contains all the citations of a given paper.

- Sina Weibo is a dataset on social media retweets. Each node is a Sina Weibo user, and an edge is a retweet. Each cascade corresponds to the retweets of one message.

Proximity of cascade subgraphs

Datasets:

- AMiner is a dataset on scientific citations. A node is an author and an edge represents a citation. A cascade contains all the citations of a given paper.

- Sina Weibo is a dataset on social media retweets. Each node is a Sina Weibo user, and an edge is a retweet. Each cascade corresponds to the retweets of one message.

We split both datasets in three intervals s. t. the 1st interval gives us a global network (for PPR computation), 2nd interval is the training set and 3rd interval is the test set.

Proximity of cascade subgraphs

	AMiner		Sina Weibo		
Time period	1 year	2 years	1h	2h	1 day
DeepCas	2.764	2.946	6.978	9.544	13.284
DeepHawkes	2.088	1.790	2.403	2.368	3.714
VerseAvg	2.313	2.181	3.580	4.243	5.862
SubRank	1.984	1.809	3.354	3.797	4.818

Table 4: Mean squared error (MSE) for predicted increase in cascade size.

Conclusion

Contributions:

- We define a novel subgraph to subgraph proximity measure;
- We introduce a framework that learns comprehensive subgraphs embeddings;
- We validate our approach on several data mining tasks [1].

Our code is online! Check it at:

https://github.com/nyxpho/subrank

[1] Disclaimer: Some macarons disappeared during the experimental evaluation.

Bibliography

Adhikari, B., Zhang, Y., Ramakrishnan, N., and Prakash, B. A. (2018).
Sub2vec: Feature learning for subgraphs.
In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 170–182. Springer.

Cao, Q., Shen, H., Cen, K., Ouyang, W., and Cheng, X. (2017).
Deephawkes: Bridging the gap between prediction and understanding of information cascades.
In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1149–1158. ACM.

Grover, A. and Leskovec, J. (2016).
node2vec: Scalable feature learning for networks.
In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM.

Li, C., Ma, J., Guo, X., and Mei, Q. (2017).
Deepcas: An end-to-end predictor of information cascades.
In *Proceedings of the 26th international conference on World Wide Web*, pages 577–586.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014).
Deepwalk: Online learning of social representations.
In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015).
Line: Large-scale information network embedding.
In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077.

Tsitsulin, A., Mottin, D., Karras, P., and Müller, E. (2018).
Verse: Versatile graph embeddings from similarity measures.
In *Proceedings of the 2018 World Wide Web Conference*, pages 539–548.