network *learning*

introduction to *network science in Python* (*NetPy*)

Lovro Šubelj
University of Ljubljana
14th Dec 2021

learning *tasks*

modern *machine* *learning* with *network* data

- *node-level* *tasks*
  - node classification (e.g. finding hoaxes on Wikipedia)
  - node ranking (e.g. finding top influencers on Instagram)
  - network clustering (e.g. research areas of scientific papers)
- *edge-level* *tasks*
  - link prediction (e.g. product recommendation on Amazon)
  - strength of ties (e.g. close friends/acquaintances on Facebook)
- *graph-level* *tasks*
  - graph classification (e.g. playing strategy in football)
  - graph generation (e.g. good candidates for new drugs)
  - etc.

learning *since 2000*

use *network analysis* techniques *directly*

- *node ranking* tasks

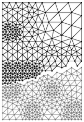
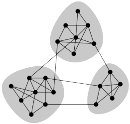
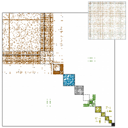
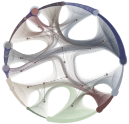
  node centrality, link analysis, graphlets, egonets etc.

- *link prediction* tasks

  link bridging, prediction indices, matrix factorization etc.

- *network clustering* tasks

  community detection, (stochastic) blockmodeling etc.

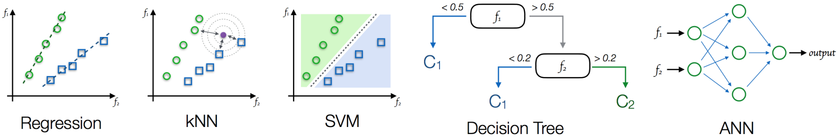





- etc.

learning *until 2010*

use *network analysis* techniques for *features*

1. generate *node/link/graph features* from *network* structure
2. feed generated *features* into *machine learning* method



but *features* are *task dependent* & redesigned every time!

for *survey* see [ZPS+16]

learning *modern*

use *machine learning* methods for *embeddings*/*directly*

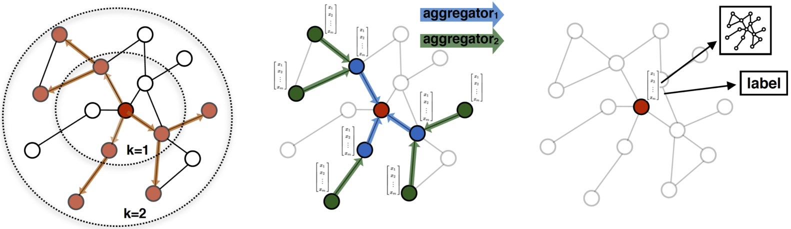
1. dimensionality reduction/*matrix factorization* (e.g. NMF)
   decomposition of adjacency matrix A or graph Laplacian L
2. *random walks* on network (e.g. node2vec [GL16], struct2vec [FRS17])
   similar nodes have *similar embeddings* independently of task
3. *graph neural networks* (e.g. GCN [KW17], GAT, GraphSAGE [HYL17])
   node/edge/graph *representations are learned* for specific task

for *survey* see [MKNŠ21]

learning *GraphSAGE*

$$h_i^0 = x_i$$

$$h_i^k = \sigma\left(W_k \cdot \mathrm{CONCAT}(h_i^{k-1}, \mathrm{AGGREGATE}_k(\{h_j^{k-1} \mid j \in \Gamma_i\}))\right)$$



1. Sample neighborhood

2. Aggregate feature information from neighbors

3. Predict graph context and label using aggregated information

for *paper* see [HYL17]

learning *references*

Daniel R. Figueiredo, Leonardo F. R. Ribeiro, and Pedro H. P. Saverese.
struc2vec: Learning node representations from structural identity.
In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1–9, 2017.

Aditya Grover and Jure Leskovec.
node2vec: Scalable feature learning for networks.
In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864, 2016.

Will Hamilton, Zhitao Ying, and Jure Leskovec.
Inductive representation learning on large graphs.
In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Thomas N. Kipf and Max Welling.
Semi-Supervised Classification with Graph Convolutional Networks.
In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, 2017.

I. Makarov, D. Kiselev, N Nikitinsky, and L. Šubelj.
Survey on graph embeddings and their applications to machine learning problems on graphs.
PeerJ Comput. Sci., 7:e357, 2021.

M. Zanin, D. Papo, P. A. Sousa, E. Menasalvas, A. Nicchi, E. Kubik, and S. Boccaletti.
Combining complex networks and data mining: Why and how.
Phys. Rep., 635:1–44, 2016.