Machine Learning on graphs. Structural Graph Embeddings

I. Makarov & L.E. Zhukov

BigData Academy MADE from VK

**Network Science**

Graph machine learning

- Node classification (attribute inference)
- Link prediction (missing/hidden links inference)
- Community detection (clustering nodes in graph)
- Graph visualization (cluster projections)

Lecture outline

Graph Embeddings

- Necessity to automatically select features
- Reduce domain- and task- specific bias
- Unified framework to vectorize network
- Preserve graph properties in vector space
- Similar nodes $\rightarrow$ close embeddings

from Leskovec et al., 2018[1]
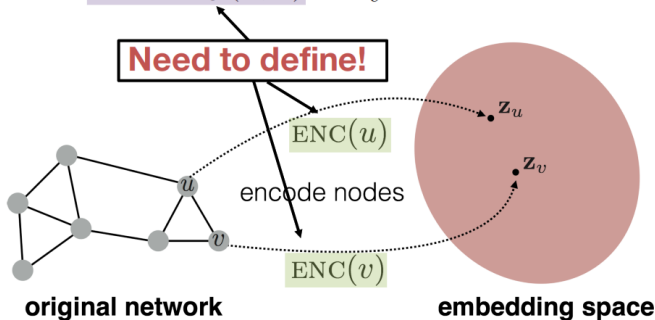
---

[1] http://snap.stanford.edu/proj/embeddings-www/
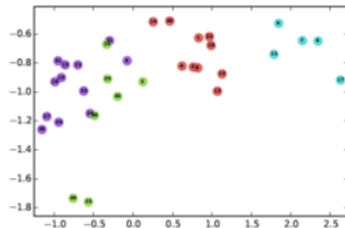
Graph Embeddings

- Define **Encoder**
- Define **Similarity**/graph feature to preserve graph properties
- Define similarity/distance in the embedding space
- **Optimize** loss to fit embedding with similarity computed on graph

Goal: $\text{similarity}(u, v) \approx \mathbf{z}_v^\top \mathbf{z}_u$

Need to define!

$\text{ENC}(u)$

$\text{ENC}(v)$

$\mathbf{z}_u$

$\mathbf{z}_v$

encode nodes

original network

embedding space

from Leskovec et al., 2018

Structural Graph Embeddings

- Embedding look-up (each node - separate vector)
- Different similarity measures (adjacency, common neighbours, distances, exact function, etc.)
- Quadratic optimization for MSE loss
- Fast models via random walks

First-order Proximity

- Similarity between u and v is A_{uv}
- MSE Loss
- Variant of Matrix Decomposition

$$\mathcal{L} = \sum_{(u,v) \in V \times V} \| \mathbf{z}_u^\top \mathbf{z}_v - \mathbf{A}_{u,v} \|^2$$

loss (what we want to minimize)

sum over all node pairs

embedding similarity

(weighted) adjacency matrix for the graph

from Leskovec et al., 2018

First-order Proximity

- Pros:
  - Use SGD for scalable optimization
  - Matrix factorization (SVD) or decomposition (QR) may be applicable
- Cons:
  - Quadratic complexity
  - Large embeddings space
  - No indirect graph properties are preserved

Multi-order Proximity

- Similarity of neighborhoods of u and v via indices or k-hop paths
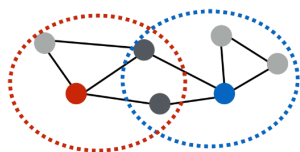- Direct optimization of exact similarity metric
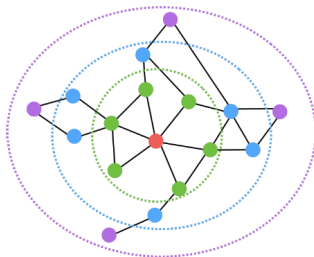
- **Red:** Target node
- **Green**: 1-hop neighbors
  - $\mathbf{A}$ (i.e., adjacency matrix)
- **Blue:** 2-hop neighbors
  - $\mathbf{A}^2$
- **Purple:** 3-hop neighbors
  - $\mathbf{A}^3$

$$\mathcal{L} = \sum_{(u,v) \in V \times V} \| \mathbf{z}_u^\top \mathbf{z}_v - \mathbf{A}_{u,v}^k \|^2$$

from Leskovec et al., 2018

Multi-order Proximity

- Similarity score S_{uv} as Jaccard/Common Neighbours, etc. (HOPE)

$$\mathcal{L} = \sum_{(u,v) \in V \times V} \| \underbrace{\mathbf{z}_u^\top \mathbf{z}_v}_{\substack{\text{embedding} \\ \text{similarity}}} - \underbrace{\mathbf{S}_{u,v}}_{\substack{\text{multi-hop network similarity} \\ \text{(i.e., any neighborhood} \\ \text{overlap measure)}}} \|^2$$

- Weighted k-hop paths with different k (GraRep)

$$\tilde{\mathbf{A}}_{i,j}^k = \max \left(\log \left(\frac{(\mathbf{A}_{i,j}/d_i)}{\sum_{l \in V} (\mathbf{A}_{l,j}/d_l)^k} \right)^k - \alpha, 0 \right)$$

node degree   constant shift

from Leskovec et al., 2018

- Even worse complexity

Random Walks

- Similarity between u and v is probability to co-occur on a random walk
- Sample each vertex u neighborhood $N_R(u)$ (multiset) by short random walks via strategy R
- Optimize similarity considering independent neighbor samples via MLE (remind Word2Vec)

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

from Leskovec et al., 2018

Random Walks

- $P(v|z_{\mathbf{u}})$ is approximated via softmax over similarity $z_{\mathbf{u}}^T \cdot z_{\mathbf{v}}$

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log \left(\frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)} \right)$$

- Problem in second Σ over all nodes
- Hard to find optimal solution

Negative Sampling

- Use *Negative Sampling* to approximate denominator

$$\log \left(\frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)} \right)$$

$$\approx \log(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)) - \sum_{i=1}^{k} \log(\sigma(\mathbf{z}_u^\top \mathbf{z}_{n_i})), n_i \sim P_V$$

random distribution over all nodes

from Leskovec et al., 2018

- Sample in proportion to node degree
- Experiment with k to impact negative prior and robustness
- No need to sample non-connected edges — same as random

Random Walk Strategies

- Finite unbiased random walks (DeepWalk)
- 1-hops & 2-hops for half of the embedding, arbitrary random walks for the second half of the embedding (LINE)
- Diffusion for sampling (Diff2Vec)
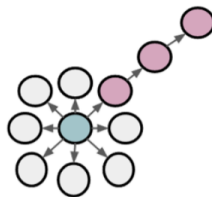- Biased random walks combining BFS and DFS (Node2Vec)

DeepWalk & LINE

- DeepWalk: Unbiased random walks with fixed length and number
- LINE: Combination of first-order and second-order proximity aggregation via concatenation

(a) GF   (b) DeepWalk   (c) LINE(2nd)

Node2Vec
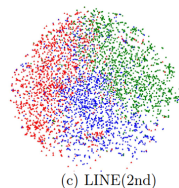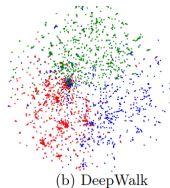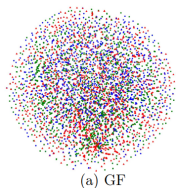
- BFS samples local neighborhood, DFS goes for global features
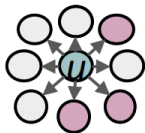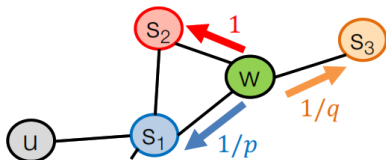

BFS: DFS:

- Two parameters p and q to control sampling
- Second order Markov process

Node2Vec

- Parameters are multiplied by edge weights and normalized for random walk probabilities
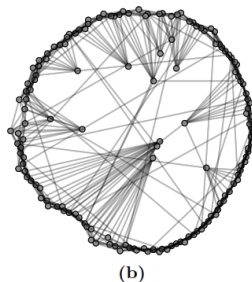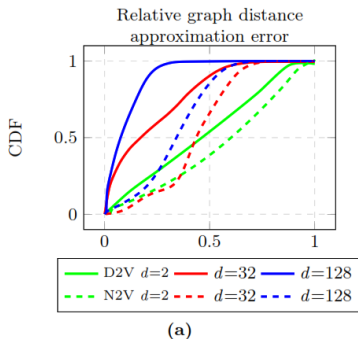

$1/p, 1/q, 1$ are unnormalized probabilities

p, q model transition probabilities

- p ... return parameter
- q ... "walk away" parameter

- Sample node according to supervised diffusion
- Efficient initialization and [possibility to control clustering parameters]

(a)

(b)

Grarep & Walklets

- Grarep: Approximate normalized A^k efficiently
- Walklets: approximate attention over k-distance neighbors
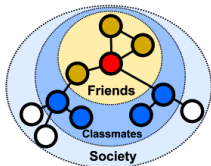
(a) A student (in red) is a member of several increasing larger social communities.

(b) WALKLETS Fine Representation

(c) WALKLETS Coarse Representation

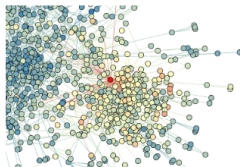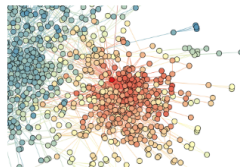- GEMSEC model cluster information adding regularization term over community labels given number of classes
- parameter γ balance cluster error and cluster structure given model hyper-parameters

(a) DeepWalk (b) GEMSEC
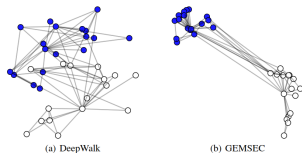
$$\mathcal{L} = \underbrace{\sum_{v \in V} \left[\ln \left(\sum_{u \in V} \exp(f(v) \cdot f(u)) \right) - \sum_{n_i \in N_S(v)} f(n_i) \cdot f(v) \right]}_{\text{Embedding cost}}$$

$$+ \gamma \cdot \underbrace{\sum_{v \in V} \min_{c \in C} \|f(v) - \mu_c\|_2}_{\text{Clustering cost}}. \tag{5}$$

M-NMF

- Modularized Nonnegative Matrix Factorization (M-NMF) incorporates first-second order proximity together with community structure.
- $S = S_1 + \eta S_2$, S_i goes for i-th order proximity.
- C represents community structure.
- H represents nodes community labels.
- B stands for $\left(A_{ij} - \frac{k_i k_j}{2m} \right)$ matrix in modularity function.
- U is our embedding.

$$\min_{\mathbf{M},\mathbf{U},\mathbf{H},\mathbf{C}} \|\mathbf{S} - \mathbf{M}\mathbf{U}^T\|_F^2 + \alpha \|\mathbf{H} - \mathbf{U}\mathbf{C}^T\|_F^2 - \beta\, tr(\mathbf{H}^T\mathbf{B}\mathbf{H})$$

$$s.t., \mathbf{M} \geqslant 0, \mathbf{U} \geqslant 0, \mathbf{H} \geqslant 0, \mathbf{C} \geqslant 0, tr(\mathbf{H}^T\mathbf{H}) = n,$$

HOPE

- HOPE is specific asymmetric transitivity preserving graph embedding.
- Asymmetric similarity measures can be formulated as $S = M_g^{-1} M_l$.
- Katz index refers to $M_g = I - \beta A$, $M_l = \beta A$.
- Rooted PageRank can be stated as
  $M_g = I - \alpha P$, $M_l = (1 - \alpha)P$, $P = D^{-1}A$.
- Common neighbors is represented by $M_g = I$, $M_l = A^2$.
- Adamic-Adar with $M_g = I$, $M_l = ADA$.
- Generalized SVD and directly estimate matrices M_g and M_l.

Other models

- Structural Deep Network Embedding (SDNE) uses autoencoder via Laplacian eigenmaps.
- Watch Your Step: Learning Node Embeddings via Graph Attention - attention on random walks aggregation without use on inference
- Metapath2vec adapts random walks on heterogeneous networks
- Discriminative Deep Random Walk (DDRW) includes label regularization to incorporate classification task

- GraphWave learns heat wavelet diffusion patterns. Nodes with similar neighbors will have similar GraphWave embeddings.
- $O(|E|)$ performance
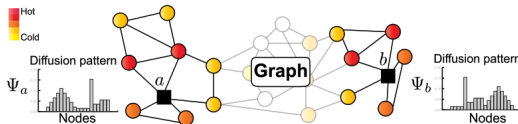
VERSE for Structural Similarity

- VERSE uses conditional probability not on the embedding, but on the similarity rank.
- $O(|E|)$ performance

$$sim_{\mathrm{E}}(v, \cdot) = \frac{\exp(W_v W^{\top})}{\sum_{i=1}^{n} \exp(W_v \cdot W_i)}$$

$$\mathcal{L} = - \sum_{v \in V} sim_{\mathrm{G}}(v, \cdot) \log(sim_{\mathrm{E}}(v, \cdot))$$

$$\mathcal{L}_{NCE} = \sum_{\substack{u \sim \mathcal{P} \\ v \sim sim_{\mathrm{G}}(u, \cdot)}} \bigg[\log \Pr_W(D = 1 | sim_{\mathrm{E}}(u, v)) +$$

$$s\mathbb{E}_{\widetilde{v} \sim Q(u)} \log \Pr_W(D = 0 | sim_{\mathrm{E}}(u, \widetilde{v})) \bigg]$$

Short conclusion for structural Graph Embeddings

- Random walks are powerful tool for fast network embedding
- Proximity-aware embeddings, random walks can be modeled in terms of each other (and even deep neural networks !)
- complexity and space are important to choose the embedding model
- provided models are used for transductive learning only, inductive learning require additional regularizations and local optimizations
- large graphs are hard to fit with handcrafted sampling strategies
- no clear way to support features

References

- B. Perozzi, R. Al-Rfou, and S. Skiena. "Deepwalk: Online learning of social representations." In Proceedings of the 20th ACM SIGKDD international conference , pp. 701-710. 2014.
- J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. "Line: Large-scale information network embedding." In Proceedings of the 24th WWW international conference , pp. 1067-1077. 2015.
- A. Grover and J. Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference, pp. 855-864. 2016.

References

- H. Cai, V.W. Zheng, and K.C.C. Chang. "A comprehensive survey of graph embedding: Problems, techniques, and applications." IEEE Transactions on Knowledge and Data Engineering 30, no. 9: 1616-1637, 2018
- Makarov, Ilya, Dmitrii Kiselev, Nikita Nikitinsky, and Lovro Subelj. "Survey on graph embeddings and their applications to machine learning problems on graphs." PeerJ Computer Science 7 (2021).