Learning Community Embedding with Community Detection and Node Embedding on Graphs

By Cavallari et al.

EGOR DMITRIEV, Utrecht Univeristy, The Netherlands

1 GOALS

- Guided by the closed loop insight, we propose ComE, a novel Community Embedding framework that **jointly solves community embedding, community detection and node embedding together**.
- Propose: a scalable inference algorithm which complexity of O(|V | + |E|), is often lower than the existing higher-order proximity-aware methods

2 PRELIMINARIES

- Graph $G = (V, E)$

3 CHALLENGES

- Community: Is a group of densely connected nodes
  - Embedding is expected to characterize how its member nodes distribute in the low-dimensional space
  - Define it as a distribution
- Pipeline approach is suboptimal:
  - Because its community detection is independent of its node embedding.
- Most existing node embeddings are not aware of community structure

4 PREVIOUS WORK / CITATIONS

- Don't try to directly detect communities
- Often: don't jointly optimize node embedding and community detection together
- SEA:
  - Construct similarity matrix from spectral GCN embeddings
  - Input it into AE for reconstruction
  - Resulting node embeddings are fed in K-means clustering
  - Shown to perform better than spectral clustering
- M-NMF:
  - Constructs modularity matrix and applies non-negative matrix factorization to learn node embeddings ans community detection together
  - Represents each community as a vector!
- **This Work:**
  - Motivated by Gaussian Mixture Models (GMM)
  - Tries to directly detect communities

5 DEFINITIONS

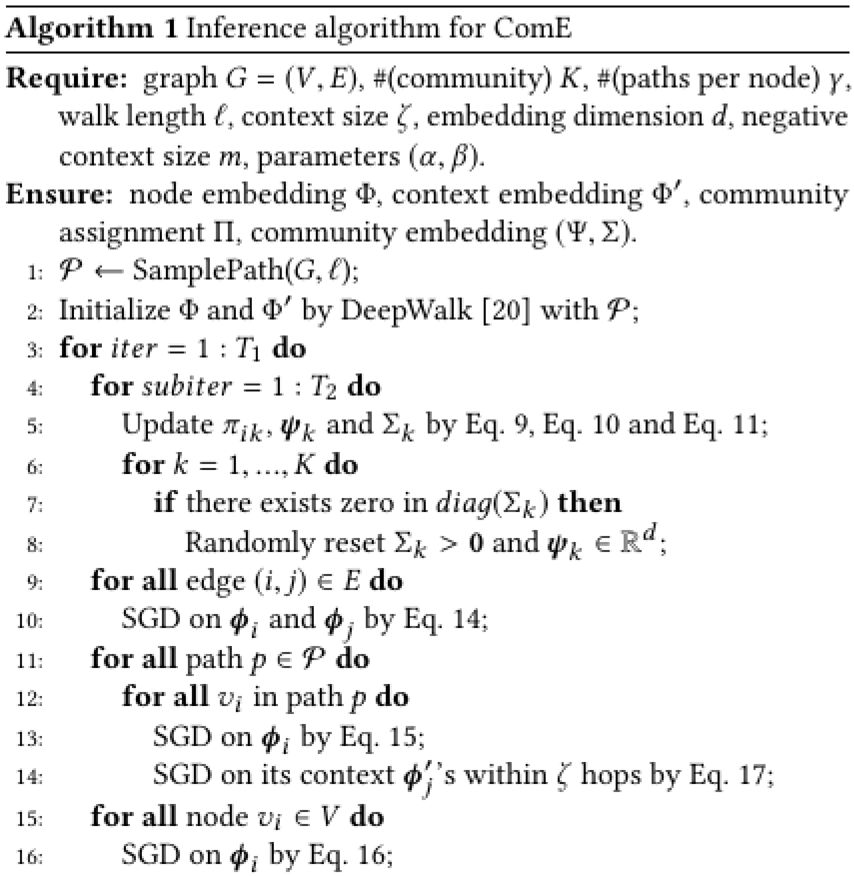
Notation	Description
$G(V, E)$	Graph G, nodes V and edges E
ℓ	Length of each random walk path in sampling
γ	Number of random walks for each node in sampling
ζ	Context size
m	Negative context size
$\boldsymbol{\phi}_i \in \mathbb{R}^d$	Node embedding of node i
$\boldsymbol{\phi}'_i \in \mathbb{R}^d$	Context embedding of node i
$\mathcal{N}(\psi_k, \Sigma_k)$	Community embedding of community k
$\psi_k \in \mathbb{R}^d$	Community embedding k 's mean vector
$\Sigma_k \in \mathbb{R}^{d \times d}$	Community embedding k 's covariance matrix
$\pi_{ik} \in [0, 1]$	Community membership of node i to community k
$P_n(\cdot)$	Negative sampling probability
K	Number of communities on G
α	Trade-off parameter for context embedding
β	Trade-off parameter for community embedding
a	graph's average degree

- **Community detection**: aims to discover groups of nodes on a graph, such that the intra-group connections are denser than the intergroup ones
  - Desired output:
    - Node embedding $\Phi_i \quad \forall v_i \in V$
    - Community membership: $\pi_{ik} \quad \forall v_i \in V$
    - Community embeddings: (Ψ_k, Σ_k)
- **Community embedding**: of community k in d-dimensional space
  - is Multivariate Gaussian distribution: $\mathcal{N}(\Psi_k, \Sigma_k)$

6 OUTLINE / STRUCTURE

- Community Detection:
  - Considering v_i and embeddings ϕ_i as generated by multivariate Gaussian distribution from a community $z_i = k$, then for all nodes v we have likelihood:
    - $\prod_{i=1}^{|V|} \sum_{k=1}^{K} p\left(z_i = k\right) p\left(v_i \mid z_i = k; \boldsymbol{\phi}_i, \psi_k, \Sigma_k\right)$ (eq 1) where
      - $p(z_i = k)$: probability of node v_i belonging to community k (π_{ik})
    - Rewritten: $p\left(v_i \mid z_i = k; \phi_i, \psi_k, \Sigma_k\right) = \mathcal{N}\left(\phi_i \mid \psi_k, \Sigma_k\right)$ where
      - (Ψ_k, Σ_k) are unknown (optimization target)
    - Optimizing achieves both community detection and embedding
- **Node embedding**:
  - **First Order Proximity**: Optimizing for direct distance of two nodes
    - $O_1 = -\sum_{(v_i, v_j) \in E} \log \sigma\left(\boldsymbol{\phi}_j^T \boldsymbol{\phi}_i\right)$
  - **Second Order Proximity**: Two nodes sharing many contexts should have similar embedding
    - Usually through negative sampling
      - $\Delta_{ij} = \log \sigma\left(\boldsymbol{\phi}_j'^T \boldsymbol{\phi}_i\right) + \sum_{t=1}^{m} \mathbb{E}_{v_l \sim P_n(v_l)}\left[\log \sigma\left(-\boldsymbol{\phi}_l'^T \boldsymbol{\phi}_i\right)\right]$
- **Higher-order proximity**: (defined for community detection / embeddings task)
  - two nodes sharing a community are likely to have similar embedding

  - $O_3 = -\frac{\beta}{K} \sum_{i=1}^{|V|} \log \sum_{k=1}^{K} \pi_{ik} \mathcal{N}\left(\phi_i \mid \psi_k, \Sigma_k\right)$
- To close the loop the objective is unified:
  - First, second and higher order proximity should be minimized
    * $\mathcal{L}\left(\Phi, \Phi', \Pi, \Psi, \Sigma\right) = O_1(\Phi) + O_2\left(\Phi, \Phi'\right) + O_3(\Phi, \Pi, \Psi, \Sigma)$
  - Optimization target:
    * $(\Phi^*, \Phi'^z, \Pi^*, \Psi^*, \Sigma^*) \leftarrow \underset{\forall k, \mathrm{diag}(\Sigma_k)>0}{\arg\min} \mathcal{L}\left(\Phi, \Phi', \Pi, \Psi, \Sigma\right)$
    * Note that Gaussian component can collapse making diag become zero and 0_3 negative infinity
- Inference:
  - Two parts:
    * Iteratively optimizing (Π, Ψ, Σ) with a constrained minimization given (Φ, Φ')
      · Constaints: $diag(\Sigma_k) > 0 \quad \forall k \in K$
      · Expectation Maximization algorithm (iteratively update params)
    * Optimizing (Φ, Φ') with an unconstrained minimization given (Π, Ψ, Σ).
  - Initialize (Φ, Φ') with deepwalk results
    * Optimize $\mathcal{L}\left(\Phi, \Phi', \Pi, \Psi, \Sigma\right)$ using SGD



Algorithm 1 Inference algorithm for ComE

Require: graph $G = (V, E)$, #(community) K, #(paths per node) γ, walk length ℓ, context size ζ, embedding dimension d, negative context size m, parameters (α, β).

Ensure: node embedding Φ, context embedding Φ', community assignment Π, community embedding (Ψ, Σ).

$\mathcal{P} \leftarrow$ SamplePath(G, ℓ);
Initialize Φ and Φ' by DeepWalk [20] with $\mathcal{P}$;
for $iter = 1 : T_1$ **do**
  for $subiter = 1 : T_2$ **do**
    Update π_{ik}, $\boldsymbol{\psi}_k$ and Σ_k by Eq. 9, Eq. 10 and Eq. 11;
    for $k = 1, \dots, K$ **do**
      if there exists zero in $diag(\Sigma_k)$ **then**
        Randomly reset $\Sigma_k > \mathbf{0}$ and $\boldsymbol{\psi}_k \in \mathbb{R}^d$;
  for all edge $(i, j) \in E$ **do**
    SGD on $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ by Eq. 14;
  for all path $p \in \mathcal{P}$ **do**
    for all v_i in path p **do**
      SGD on $\boldsymbol{\phi}_i$ by Eq. 15;
      SGD on its context $\boldsymbol{\phi}_j'$'s within ζ hops by Eq. 17;
  for all node $v_i \in V$ **do**
    SGD on $\boldsymbol{\phi}_i$ by Eq. 16;

Fig. 1. Screenshot_20211102_150521

7 EVALUATION

- average degree can impact performance due to length and the window size limitation of random walks

8 DISCUSSION

- The part / constraint enforcement for zero checking $diag(\Sigma_k)$ is not really that convincing

9 CODE

- https://github.com/vwz/ComE

10 RESOURCES

- ...

11 NOTES

- What is a modularity matrix?