Community-aware dynamic network embedding by using deep autoencoder

By Ma et al

1 GOALS

- Propose a Community-aware Dynamic Network Embedding method (CDNE)
  - aiming to learn the low-dimensional representations of nodes over time while preserving the global node structures, local link structures, and continuous community dynamics
  - validate the superiority of CDNE over several state-of-the-art DNE methods

2 PRELIMINARIES

- **Network Embedding**: technique for learning the low-dimensional representation of entities while preserving their properties
- Generally, the **high-dimensional** structures are represented by:
  - the **microscopic structures** such as the link proximity [41], cycles, subspaces [44], paths [32] and multilayered links [22]
  - the **macroscopic structures** such as the community structures and subgraphs
- Many real-world networks such as collaboration networks, social networks and biological networks are naturally dynamic, with the emergence and disappearance of nodes and edges over time
-

3 CHALLENGES

- Smooth dynamic assumption in [7,11,12,14] does not always hold as the **microscopic node and link structures of some systems may evolve sharply**
-

4 PREVIOUS WORK / CITATIONS

- Some dynamic NE (DNE) methods have been proposed under the assumption that the **microscopic node and link structures** constantly change over time: (AKA **dynamic node representation learning**)
  - Goyal et al. [12,14] proposed to learn the low-dimensional representations of nodes incrementally:
    - Representations at current step depending those on previous time steps
    - Proposed Dyn2vec and DySAT, which use a self-attention technique and a vector recurrent technique to learn the potential structures of dynamic networks at **multiple non-linear time stamps**, respectively [11].
  - Du et al. [7] learned the smooth evolution of the most influential nodes
  - DHPE [50] imposed some extra feature information into dynamic networks
  - Drawbacks:

Author's address:

    * This assumption neglects the dynamics of the macroscopic community structures in dynamic network
    * Microscopic node and link structures of some systems may evolve sharply
- **This Work:**
  - We present a novel dynamic network embedding method (called as CDNE)
  - With **consideration of the dynamics** of the **microscopic** node structures **and macroscopic** community structures
  - To preserve the dynamics of communities, we present a **community-aware smoothing technique** and **adopt a *temporal regularizer***.
  - We use a combination of the ***first-order proximity*** and the ***second-order proximity*** of nodes in dynamic networks
  - We adopt a ***stacked deep autoencoder*** algorithm to learn the low-dimensional representations of nodes
  - Intuition:
    * Communities generally have a smoother evolution than nodes and links in dynamic systems
    * Embedding representations of a community at adjacent time should be close to each other in the low-dimensional space

5 DEFINITIONS

- **Dynamic network**: $\mathcal{G}$ is a series of network snapshots
- **Adjacency matrix**: A^a Represents a snapshot at time a
- **Community Structure**: A community in a dynamic network consists of a set of nodes, such that the nodes in the community are densely linked with each other whereas they are sparsely connected with the other nodes *at all time stamps*
  - communities have a smoother evolution than nodes and links in dynamic systems
- **Dynamic Network Embedding (DNE)**:
  - Finds a time-series of mappings $\mathbf{F} = \{\mathbf{F^1}, \mathbf{F^2}, ...\mathbf{F^t}\}$, and each mapping $F^a : S^a \rightarrow H^a$, learns the latent d dimensional representations H^a of nodes at ath time stamp such that the learned representations can effectively preserve both the **structural information** $S = \{S^1, S^2, ..., S^t\}$ and the evolutionary pattern in $\mathcal{G}$.

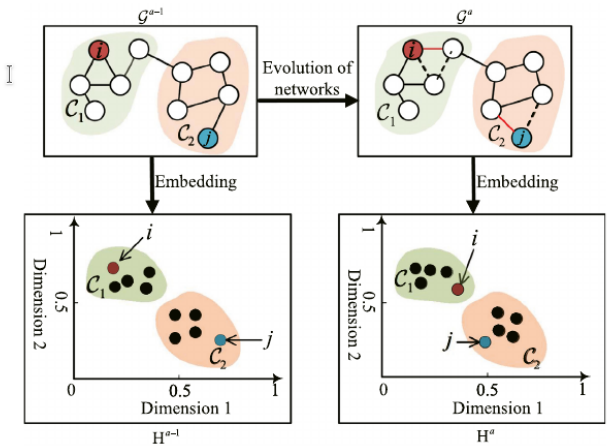

Fig. 1. Schematic illustration of the nodes and communities of a toy network with 10 nodes and 2 time stamps in both network dimensions $\mathcal{G}$ and embedded dimensions H. Each set represents a community, and there are two communities in the toy network. The dotted links denote the disappeared links while the full links marked with red colors are the appeared links at the next time stamp a, respectively.

6 OUTLINE / STRUCTURE

6.1 Global structure preservation

- Global structure of a network reflects the **similarities of nodes** which are usually evaluated by their link structures.
- Evaluating by link structures neglects the similarities of two unlinked nodes
- **Use a combination** of the **first-order proximity** S^1 and the **second-order proximity** S^2 to represent the global structure of networks
  - **first-order proximity** S^1: evaluates the number of direct neighbor of nodes
  - **second-order proximity** S^2: measures the similarity of the common neighbors of node
  - Global structure S^a: $S^a = S^{1,a} + \lambda S^{2,a}$
- **Global Reconstruction Error**: $\mathrm{L}_g^a = \sum_{i=1}^{n} \left\| \hat{S}_{i.}^a - S_{i.}^a \right\|_2^2$ where
  - Preserves the global structures by adopting objective of a stacked deep autoencoder
  - S: **true global structures**
  - $\hat{S} = D(F(S))$: structures **decoded by the decoder** D from the low-dimension representations generated by the encoder F for the structures S.

6.2 Local Structure Preservation

- Local structure of a node is the neighbors of the node
- From the view of social ***homophily***, the neighbors of an individual i are a set of individuals that are highly connected with i.
- **Local Loss**: $\mathrm{L}_l^a = \sum_{e_{ij} \in \mathcal{E}^a} \left\| \mathrm{H}_i^a - \mathrm{H}_{j.}^a \right\|_2^2$ where
  - $H_i^a.$ is the low-dimension representation of node i at ath timestamp

6.3 Evolution community preservation

- **Community** in a dynamic network G is composed of a set of nodes which are **densely linked** with each other and **have similar structure functions** at most time stamps
  - **Microscopic** node structures temporally change over time
  - **Macroscopic** community structures smoothly evolve over time
  - Node may sharply change links, but its community evolves smoothly
- **Community Evolution Loss**: $\mathrm{L}_c^a = \begin{cases} \sum_{k=1}^{q} \sum_{i \in C_k} \left\| \mathrm{H}_i^a - \mathrm{H}_{c_k}^{a-1} \right\|_2^2 & \text{if } a > 1 \\ 0 & \text{if } a = 0 \end{cases}$
  - Minimizes the Euclidean distance between nodes and their communities in the low-dimensional representations at adjacent time stamps
  - Allows for **maintaining community stability**
  - $\mathrm{H}_{c_k}^a$: is the low-dimensional representation of community C_k at ath time stamp
    - $\mathrm{H}_{C_k}^a = \frac{\sum_{i \in C_k} \mathrm{H}_{i.}^a}{|C_k|}$ is average of its members' representations
- **Trade-off between the sensibility and stability** (Small Scale vs Large Scale communities)
  - In the embedded space, a **node is closer to its small-scale community than its large-scale community**
  - Small-scale community is more sensitive to the evolution than a large-scale community

- **Community Evolution Loss (Hierarchical)**: $\mathbf{L}_c^a = \begin{cases} \sum_{k=1}^{q} \sum_{j=1}^{q_k} \sum_{i \in C_k^j} \left\| \mathrm{H}_{i.}^a - \mathrm{H}_{C_k^j}^{a-1} \right\|_2^2 & \text{if } a > 1 \\ 0 & \text{if } a = 0 \end{cases}$

  where
  - Community: $C_k = \left\{C_k^1, C_k^2, \ldots, C_k^{q_k}\right\}$
  - q_k is the number of small scale communities
  - $\mathrm{H}_{C_k^j}^a = \frac{\sum_{i \in C_k^j} \mathrm{H}_i^a}{\left|C_k^j\right|}$
- Number of sub communities: q_k
  - Determined by w (hyperparameter) which controls the size of small communities
- Calculating the loss requires already having communities:
  - **Requires having some prior knowledge about community structures**
  - Communities are **not known a priori**
  - Therefore they are **initialized using existing CD algorithms** (classical)
    - Large scale communities: Genlouvin (requires no priors)
      - More rough but very fast
    - Small scale communities: k-means (prior is w community size)
      - More fine tuned

6.4 Objective

- Given a dynamic network $\mathcal{G}$ and dimension size d
  - Find optimal encoders: $\mathbf{F} = \{\mathbf{F^1}, \mathbf{F^2}, \ldots \mathbf{F^t}\}$
    - Which maps structure S^a to a (low) d-dimensional representation H^a
  - Find optimal decoders: $\mathbf{D} = \{\mathbf{D^1}, \mathbf{D^2}, \ldots \mathbf{D^t}\}$
    - Which maps structure H^a to a high-dimensional network representation $\hat{S}^a$
- Unified loss: $\mathbf{L}^a = \mathbf{L}_g^a + \alpha \cdot \mathbf{L}_l^a + \beta \cdot \mathbf{L}_c^a$
  - α affects the performance of our CDNE on the **prediction** tasks
  - β has significant impacts on the **stabilization** tasks
  - $\mathbf{L}_l^a$ and $\mathbf{L}_c^a$ are referred to as regularizers
  -

6.5 Architecture

- Employs quite simplistic Spectral GAE architecture
- Encoder: $\mathrm{H}_{i.}^{a,o} = \mathbf{F}^{a,o}\left(\mathrm{H}_{i.}^{a,o-1}\right) = \sigma\left(\mathrm{W}^{a,o} \cdot \mathrm{H}_{i.}^{a,o-1} + \mathrm{b}^{a,o}\right)$
  - Note that $\mathrm{H}_{i.}^{a,0} = S^a$
  - o is the layer number
  - $S^a = \hat{S}^a = \left[\hat{S}_{ij}^a\right]_{n \cdot n} \in \mathbb{R}^{n \times n}$ kinda Laplacian matrix but different
  - Uses **Unified Loss** L^a
- Decoder: $\hat{S}_{i.}^{a,o} = \mathbf{D}^{a,o}\left(\mathrm{H}_{i.}^{a,o}\right) = \sigma\left(\hat{\mathrm{W}}^{a,o} \cdot \mathrm{H}_{i.}^{a,o} + \hat{\mathrm{b}}^{a,o}\right)$
  - Maps back to the "kinda" Laplacian matrix
  - Uses only **Reconstruction Loss** L_g^a

7 EVALUATION

- Datasets:

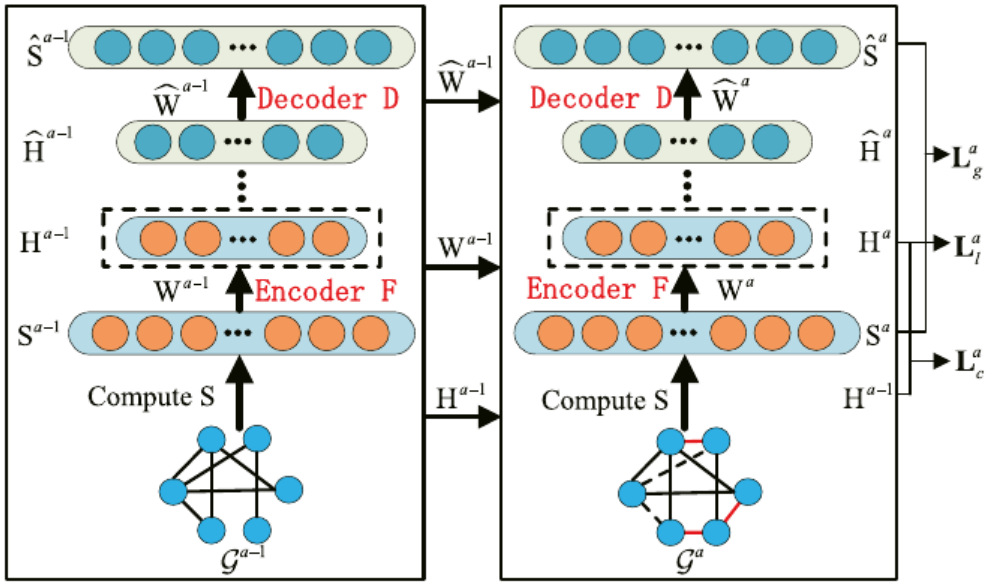

Fig. 3. CDNE: Community-aware Dynamic Network Embedding at ath time stamp.

  - Synthetic:
    * SYN **?**]
    * SBM **?**]
  - Real:
    * ia-contacts, ia-email, ia-enron, ia-stackexch - **?**]
    * TCGA, ca-cit-HepTh
    * soc-wiki-elec - Wikipedia Adminitrat
    * football
- Baselines: - Mainly Dynamic Network Embedding models
  - SDNE - structural deep autoencoder for static networks
  - M-NMF - modularized nonnegative matrix factorization (NMF)
  - SAGE - GraphSAGE - not
  - DynGEM - It uses a deep autoencoder to solve the DNE problem in dynamic networks
  - Dyn2vecAE - Capturing network dynamics using dynamic graph representation learning
- Criteria:
  - **Network reconstruction**: It is used to test the performance of NE methods on reconstructing the link structures of networks
    * Average reconstruction precision (pr)
  - **Link prediction**:
    * For dynamic networks, it is adopted to predict the existence of links in the next time stamp

  - **Network stabilization**:
    * Performance of DNE methods on the stabilization of embedding
    * Dynamic network should have similar evolutionary patterns in both the learned low-dimensional representation and the network representation over time
  - **Community stabilization**:
    * Stability of communities in dynamic networks on the embedded low-dimensional representations.

8 DISCUSSION

- A lot of small mistakes in the paper (as if it is a prepublication - it is not)
- There is no source code
- The baselines are not DCD algorithms but Dynamic Network Embedding algorithms
- Community memberships can't change?
- This, and **?**]
  - Use datasets with very few nodes
  - Possibly because of the gigantic matrices for reconst

9 CODE

- …

10 RESOURCES

- …